

**Система проведения соревнований ejudge**  
**Справочное руководство**

Москва — 2004

© Чернов Александр Владимирович, 2003-2004

Каждый имеет право воспроизводить, распространять и/или вносить изменения в настоящий Документ в соответствии с условиями GNU Free Documentation License, Версии 1.1 или любой более поздней версией, опубликованной Free Software Foundation. Данный документ не содержит неизменяемого текста, текста, помещаемого на первой и последней страницах обложки. Копия настоящей Лицензии включена в раздел под названием «GNU Free Documentation License». Неофициальный перевод лицензии на русский язык включён в раздел «Перевод на русский язык Лицензии GNU на свободную документацию».

# Оглавление

<b>1</b>	<b>Общая архитектура системы</b>	<b>7</b>
<b>2</b>	<b>Конфигурационные файлы</b>	<b>8</b>
2.1	Полномочия пользователей	8
2.2	Ограничение доступа по IP-адресам	10
2.2.1	Ограничения IP-адресов в формате XML	11
2.2.2	Элемент <code>access</code>	12
2.2.3	Элемент <code>ip</code>	12
2.2.4	Пример использования элемента <code>access</code>	13
2.2.5	Ограничения IP-адресов в текстовом формате	13
2.3	Локализация системы	14
2.3.1	Включение поддержки языкового окружения	14
2.3.2	Установка каталога сообщений	15
2.3.3	Поддержка кодировок символов	15
2.4	Общий конфигурационный файл <code>ejudge.xml</code>	16
2.4.1	Общая структура	17
2.4.2	<code>Ter config</code>	17
2.4.3	<code>Ter cap</code>	18
2.4.4	<code>Ter caps</code>	18
2.4.5	<code>Ter contests_dir</code>	19
2.4.6	<code>Ter email_program</code>	19
2.4.7	<code>Ter ll0n_dir</code>	20
2.4.8	<code>Ter map</code>	20
2.4.9	<code>Ter register_email</code>	21
2.4.10	<code>Ter register_url</code>	21
2.4.11	<code>Ter serve_path</code>	22
2.4.12	<code>Ter run_path</code>	22
2.4.13	<code>Ter socket_path</code>	22
2.4.14	<code>Ter user_map</code>	24
2.4.15	<code>Ter userdb_file</code>	24
2.5	Конфигурационный файл <code>register.xml</code>	25
2.5.1	<code>Ter register_config</code>	25
2.5.2	<code>Ter access</code>	27
2.5.3	<code>Ter ip</code>	27
2.5.4	<code>Ter ll0n_dir</code>	28
2.5.5	<code>Ter contests_dir</code>	28
2.5.6	<code>Ter socket_path</code>	29
2.6	Конфигурационный файл <code>users.xml</code>	30

2.6.1	Ter users_config	30
2.6.2	Ter access	32
2.6.3	Ter ip	32
2.6.4	Ter l10n_dir	33
2.6.5	Ter contests_dir	33
2.6.6	Ter socket_path	34
2.7	Конфигурационный файл master.cfg	34
2.7.1	Параметр allow_deny	35
2.7.2	Параметр allow_from	35
2.7.3	Параметр deny_from	35
2.7.4	Параметр charset	35
2.7.5	Параметр enable_l10n	36
2.7.6	Параметр l10n_dir	36
2.7.7	Параметр socket_path	36
2.7.8	Параметр contests_dir	37
2.8	Конфигурационный файл judge.cfg	37
2.8.1	Параметр allow_deny	38
2.8.2	Параметр allow_from	38
2.8.3	Параметр deny_from	38
2.8.4	Параметр charset	38
2.8.5	Параметр enable_l10n	39
2.8.6	Параметр l10n_dir	39
2.8.7	Параметр socket_path	39
2.8.8	Параметр contests_dir	40
2.9	Конфигурационный файл team.cfg	40
2.9.1	Параметр allow_deny	41
2.9.2	Параметр allow_from	41
2.9.3	Параметр deny_from	41
2.9.4	Параметр charset	41
2.9.5	Параметр enable_l10n	42
2.9.6	Параметр l10n_dir	42
2.9.7	Параметр socket_path	42
2.9.8	Параметр contests_dir	43
2.9.9	Параметр show_generation_time	43
2.10	Конфигурационный файл турниров contest.xml	44
2.10.1	Элемент contest	48
2.10.2	Элемент name	52
2.10.3	Элемент name_en	52
2.10.4	Элемент register_access	52
2.10.5	Элемент users_access	53
2.10.6	Элемент master_access	53
2.10.7	Элемент judge_access	53
2.10.8	Элемент team_access	54
2.10.9	Элемент ip	54
2.10.10	Элемент field	54
2.10.11	Элементы contestants, reserves, coaches, advisors, guests	56
2.10.12	Элемент users_header_file	58
2.10.13	Элемент users_footer_file	58

2.10.14	Элемент <code>register_header_file</code>	59
2.10.15	Элемент <code>register_footer_file</code>	59
2.10.16	Элемент <code>team_header_file</code>	60
2.10.17	Элемент <code>team_footer_file</code>	60
2.10.18	Элемент <code>users_head_style</code>	61
2.10.19	Элемент <code>users_par_style</code>	62
2.10.20	Элемент <code>users_table_style</code>	63
2.10.21	Элемент <code>users_verb_style</code>	64
2.10.22	Элемент <code>register_head_style</code>	65
2.10.23	Элемент <code>register_par_style</code>	66
2.10.24	Элемент <code>register_table_style</code>	67
2.10.25	Элемент <code>team_head_style</code>	68
2.10.26	Элемент <code>team_par_style</code>	69
2.10.27	Элемент <code>register_email</code>	70
2.10.28	Элемент <code>register_url</code>	70
2.10.29	Элемент <code>team_url</code>	71
2.10.30	Элемент <code>registration_deadline</code>	71
2.11	Конфигурационный файл сервера турнира <code>serve.cfg</code>	73
2.11.1	Общая структура	73
2.11.2	Форматные подстановки	74
2.11.3	Список глобальных параметров	76
2.11.4	Глобальные параметры	82
2.11.5	Параметры задачи	162
2.11.6	Параметры языка	197
2.11.7	Параметры тестировщика	205
2.12	Файл описания теста <code>test.inf</code>	228
2.12.1	Использование	228
2.12.2	Общая структура	228
2.12.3	Конфигурационные переменные	230
<b>3</b>	<b>Разработка проверяющих программ</b>	<b>232</b>
3.1	Проверяющие программы	232
3.1.1	Параметры командной строки	232
3.1.2	Результат проверки	233
3.2	Библиотека <code>libchecker</code>	233
3.2.1	Вид проверяющей программы	234
3.2.2	Константы	235
3.2.3	Глобальные переменные	235
3.2.4	Макросы распределения памяти	238
3.2.5	Функции выдачи диагностики	239
3.2.6	Функции распределения памяти	241
3.2.7	Проверка конца файла	242
3.2.8	Закрытие файла	242
3.2.9	Чтение целых и вещественных данных	242
3.2.10	Чтение текста	243
3.2.11	Примеры	247

<b>4</b>	<b>Веб-интерфейс пользователя</b>	<b>249</b>
4.1	Интерфейс администратора турнира <b>master</b>	249
4.1.1	Конфигурационные файлы	249
4.1.2	Вход в систему	251
4.1.3	Управление турниром до его начала	254
4.1.4	Управление турниром в течение турнира	257
4.1.5	Специальные элементы управления турниром	264
4.1.6	Управление турниром после его окончания	266
4.1.7	Элементы отображения журнала посылок	268
4.1.8	Элементы управления журналом посылок	270
4.1.9	Элементы просмотра журнала сообщений	274
4.1.10	Элементы интерфейса посылки сообщений	277
4.1.11	Выражение фильтра посылок	279
4.1.12	Интерфейс управления участниками турнира	290
4.1.13	Интерфейс редактирования посылок	292
4.1.14	Просмотр исходного кода посылок	298
4.1.15	Просмотр протокола тестирования	300
4.1.16	Интерфейс просмотра сообщений	303
4.1.17	Интерфейс ответа на сообщение	305
<b>5</b>	<b>Внутренние файлы</b>	<b>308</b>
5.1	Таблица посылок	308
5.1.1	Заголовок таблицы	308
5.1.2	Запись о посылке в таблице	309
5.1.3	Статус посылки	313
<b>A</b>	<b>Лицензионные соглашения</b>	<b>321</b>
A.1	GNU Free Documentation License	321
A.1.1	Applicability and Definitions	321
A.1.2	Verbatim Copying	322
A.1.3	Copying in Quantity	322
A.1.4	Modifications	323
A.1.5	Combining Documents	324
A.1.6	Collections of Documents	325
A.1.7	Aggregation With Independent Works	325
A.1.8	Translation	325
A.1.9	Termination	325
A.1.10	Future Revisions of This License	326
A.2	Перевод на русский язык Лицензии GNU на свободную документацию	327
A.2.1	Сфера действия, термины и их определения	328
A.2.2	Копирование без внесения изменений	329
A.2.3	Тиражирование	329
A.2.4	Внесение изменений	330
A.2.5	Объединение документов	332
A.2.6	Сборники документов	332
A.2.7	Подборка документа и самостоятельных произведений	333
A.2.8	Перевод	333
A.2.9	Расторжение лицензии	333

A.2.10	Пересмотр условий лицензии . . . . .	333
A.2.11	Порядок применения условий настоящей Лицензии . . . . .	334
A.3	GNU GENERAL PUBLIC LICENSE . . . . .	335
A.3.1	Terms and conditions for copying, distribution and modification . . . . .	335
A.3.2	How to Apply These Terms to Your New Programs . . . . .	339
A.4	Перевод на русский язык GNU General Public License . . . . .	341
A.4.1	Условия воспроизведения, распространения и модификации . . . . .	342
A.4.2	Порядок применения условий настоящей Лицензии . . . . .	347
A.4.3	Примечания переводчика . . . . .	349

# Глава 1

## Общая архитектура системы

Система **ejudge** включает в себя несколько компонент, обменивающихся информацией через общие каталоги или через UNIX-сокеты. Компоненты первого типа могут быть распределёнными на несколько машин, компоненты второго типа должны выполняться на одной (серверной) машине.

В центре системы находятся две компоненты — **userlist-server** и **serve**. Компонента **userlist-server** поддерживает базу данных зарегистрированных пользователей, выполняя запросы как от пользователей (через cgi-программы **users**, **register**), так и от администратора базы (через программу **edit-userlist**). Все запросы принимаются только через локальный UNIX-сокет, поэтому клиенты могут работать только на том же самом компьютере.

Компонента **serve** выполняет функции сервера турнира, то есть принимает от пользователей информационные запросы и решения отосланные на проверку, выдаёт полученные решения на компиляцию и проверку, обновляет текущие результаты и т. д. Запросы от пользователей и администратора турнира принимаются по локальному UNIX-сокету. Как правило, эти запросы формируются cgi-программами **master** и **team**. Обмен с компонентами **compile** и **run** ведётся через общие каталоги, что позволяет размещать их на нескольких компьютерах, разделяющих общую сетевую файловую систему.

Программа **compile** компилирует поступивший ей файл с исходным текстом программы. Скомпилированная программа при успешной компиляции или список ошибок при неуспешной компиляции возвращаются серверу турнира также через общие каталоги.

Программа **run** запускает поступившую ей от сервера скомпилированную программу на заданном наборе тестов и в зависимости от режима проверки формирует ответ, который передаётся обратно серверу вместе с журналом проверки.



# Глава 2

## Конфигурационные файлы

В данном разделе описываются конфигурационные файлы системы.

### 2.1 Полномочия пользователей

Система ejudge поддерживает тонкую настройку прав выполнения всех привилегированных операций для пользователей системы. К привилегированным операциям относятся операции управления турниром (запуск, останов, просмотр и изменения результатов проверки программы и пр.) и операции управления базой данных пользователей. Настройка осуществляется с помощью указаний полномочий пользователя для выполнения операций над базой пользователей и полномочий пользователя для выполнения операций над каждым из созданных в системе турниров.

Пользователи, для которых указаны полномочия выполнения операций над базой пользователей, считаются привилегированными. При этом может быть указано пустое множество полномочий. Пользователи, для которых указаны полномочия выполнения операций над каким-либо турниром, являются привилегированными только для данного турнира, но не для системы в целом. Различие между общесистемно привилегированным пользователем (даже с пустым множеством привилегий) и всеми прочими пользователями состоит в том, что для выполнения операций над привилегированными пользователями требуются специальные привилегии.

Все привилегии системы ejudge перечислены ниже.

Название	Бит	Где	Описание
MASTER_LOGIN	0	С	Пользователь может использовать CGI-программу <a href="#">master</a> .
JUDGE_LOGIN	1	С	Пользователь может использовать CGI-программу <a href="#">judge</a> .
SUBMIT_RUN	2	С	Пользователь может посылать решения, пользуясь привилегированными CGI-программами <a href="#">master</a> или <a href="#">judge</a> .
MAP_CONTEST	3	С	Пользователь может запускать сервер турнира программу <a href="#">serve</a> .
LIST_CONTEST_USERS	4	С	Пользователь может просматривать список пользователей, зарегистрировавшихся на турнир.
<i>Продолжение на следующей странице</i>			

Название	Бит	Где	Описание
LIST_ALL_USERS	5	U	Пользователь может просматривать список всех пользователей в базе пользователей.
CREATE_USER	6	U	Пользователь может создавать новых пользователей.
GET_USER	7	CU	Пользователь может получать информацию о другом непривилегированном пользователе.
EDIT_USER	8	CU	Пользователь может редактировать информацию о другом непривилегированном пользователе.
DELETE_USER	9	U	Пользователь может удалять другого непривилегированного пользователя из базы пользователей.
PRIV_EDIT_USER	10	CU	Пользователь может редактировать другого информацию о другом привилегированном пользователе.
PRIV_DELETE_USER	11	U	Пользователь может удалять другого привилегированного пользователя из базы пользователей.
GENERATE_TEAM_PASSWORDS	12	C	Пользователь может генерировать пароли для CGI-программы <b>team</b> для всех других непривилегированных пользователей, зарегистрировавшихся на данный турнир.
CREATE_REG	13	C	Пользователь может регистрировать других непривилегированных пользователей для участия в турнире.
EDIT_REG	14	C	Пользователь может редактировать статус регистрации других пользователей в турнире.
DELETE_REG	15	C	Пользователь может удалять регистрацию других непривилегированных пользователей в турнире.
PRIV_CREATE_REG	16	C	Пользователь может регистрировать других привилегированных пользователей для участия в турнире.
PRIV_DELETE_REG	17	C	Пользователь может удалять регистрацию других привилегированных пользователей в турнире.
DUMP_USERS	18	CU	Пользователь может получать базу пользователей в CSV-формате.
DUMP_RUNS	19	C	Пользователь может получать базу попыток сдачи в CSV-формате.
DUMP_STANDINGS	20	C	Пользователь может получать результаты турнира в CSV-формате.
VIEW_STANDINGS	21	C	Пользователь может просматривать судейскую таблицу результатов, которая никогда не замораживается.
VIEW_SOURCE	22	C	Пользователь может просматривать исходный код попыток сдачи программ участников.
VIEW_REPORT	23	C	Пользователь может просматривать отчёт о тестировании программ участников.
VIEW_CLAR	24	C	Пользователь может просматривать вопросы к судьям от участников и ответы судей.
EDIT_RUN	25	C	Пользователь может произвольным образом изменять информацию о попытке участника.

*Продолжение на следующей странице*

Название	Бит	Где	Описание
REJUDGE_RUN	26	С	Пользователь может отправить попытку участника на перетестирование.
NEW_MESSAGE	27	С	Пользователь может послать сообщение произвольному участнику.
REPLY_MESSAGE	28	С	Пользователь может ответить на сообщение от участника.
CONTROL_CONTEST	29	С	Пользователь может управлять турниром (стартовать, останавливать и т. д.).
IMPORT_XML_RUNS	30	С	Пользователь может импортировать таблицу посылок турнира в XML-формате.
PRINT_RUN	31	С	Пользователь может распечатывать на принтере текст программы из-под привилегированных программ <code>master</code> или <code>judge</code> .

Полномочия пользователей для базы данных пользователей задаются в общем конфигурационном файле `ejudge.xml` с помощью элемента `cap`. В этом элементе через запятую перечисляются полномочия указанного пользователя. Пробельные символы игнорируются. Например:

```
<cap login="vasya">
  LIST_ALL_USERS,
  GET_USER,
</cap>
```

Текст в элементе `cap` может быть пустым, тогда множество полномочий пользователя пусто, но он будет считаться привилегированным пользователем. Например:

```
<cap login="petya"/>
```

Полномочия пользователя по отношению к турниру задаются в конфигурационном файле турнира `contest.xml` с помощью элемента `cap`. В элементе через запятую перечисляются полномочия указанного пользователя. Пустое множество полномочий для турнира допускается, но не имеет смысла.

## 2.2 Ограничение доступа по IP-адресам

Система `ejudge` включает в себя несколько программ: `register`, `users`, `master`, `judge`, `team`, — запускаемых как CGI-программы. Каждая из этих программ позволяет ограничивать диапазон IP-адресов клиента, которым разрешается использование CGI-программы. В текущей версии системы ограничение диапазонов возможно только с помощью указания IPv4-адресов клиентов. Ни DNS-имена, ни IP протокол версии 6 в текущей версии не поддерживаются.

Кроме того, каждый турнир может накладывать дополнительные ограничения на допустимые IP-адреса для всех CGI-программ. Эти ограничения указываются в конфигурационном файле турнира `contest.xml`.

Спецификация ограничения адреса может либо задавать конкретный IP-адрес, либо задавать адрес сети классов А, В или С. Задание подсетей в сетях этих классов в настоящее

A.B.C.D	Здесь A, B, C и D — десятичные числа от 0 до 255. В этом случае задаётся один IP-адрес.
A.B.C.	A, B и C — десятичные числа от 0 до 255. В этом случае задаётся IP-адрес сети класса C. Обратите внимание на заключительную точку в записи шаблона.
A.B.	A и B — десятичные числа от 0 до 255. В этом случае задаётся IP-адрес сети класса B. Обратите внимание на заключительную точку в записи шаблона.
A.	A — десятичное число от 0 до 255. В этом случае задаётся IP-адрес сети класса A. Обратите внимание на заключительную точку в записи шаблона.

Таблица 2.2: Допустимые спецификации ограничения IP-адреса

время не поддерживается. Допустимые спецификации ограничения IP-адреса перечислены в таблице 2.2.

Синтаксис задания IP-ограничений отличается для CGI-программ, использующих конфигурационные файлы в формате XML ([register](#), [users](#)), и для CGI-программ, использующих традиционные текстовые файлы ([master](#), [judge](#), [team](#)). Далее рассматриваются оба варианта. В конфигурационном файле турнира [contest.xml](#) ограничения задаются в формате XML.

### 2.2.1 Ограничения IP-адресов в формате XML

Спецификация ограничений на IP-адрес представляет собой список спецификаций IP-адресов, для каждого из которых указано, допустимо ли использование CGI-программы клиентом с IP-адресом, удовлетворяющим спецификации, или нет. Элементы списка задаются элементом [ip](#) XML-файла. Описание этого элемента дано ниже.

Для конфигурационных файлов программ ([register](#) и [users](#)), элементы списка находятся в элементе [access](#) (см. описание элемента [access](#) конфигурационного файла [register.xml](#) и описание элемента [access](#) конфигурационного файла [users.xml](#)). Для конфигурационного файла турнира [contest.xml](#) используются элементы [register\\_access](#), [users\\_access](#), [master\\_access](#), [judge\\_access](#), [team\\_access](#), задающие дополнительные ограничения на допустимые IP-адреса для программ [register](#), [users](#), [master](#), [judge](#) и [team](#) соответственно. Далее даётся описание XML-элемента [access](#), справедливое для всех вышеперечисленных элементов.

При работе CGI-программы список ограничений просматривается последовательно от первого элемента к последнему. Как только будет найдена первая спецификация, которой удовлетворяет IP-адрес клиента, дальнейший просмотр прекращается и выполняется действие, указанное в этой спецификации. Если IP-адрес клиента не удовлетворяет ни одной спецификации, выполняется действие по умолчанию, заданное в элементе верхнего уровня.

## 2.2.2 Элемент `access`

Имя элемента:	<code>access</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент является элементом верхнего уровня списка ограничений IP-адресов для CGI-программ.

### Атрибут `default`

Имя атрибута:	<code>default</code>
Содержится в:	<code>access</code>
Тип значения:	<i>allow</i> или <i>deny</i>
Значение по умолчанию:	<i>deny</i>
Может отсутствовать:	<i>да</i>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента не перечислен в списке IP-адресов. Если атрибут установлен в *allow*, такие IP-адреса считаются допустимыми, а если атрибут установлен в *deny*, доступ с таких IP-адресов запрещается.

## 2.2.3 Элемент `ip`

Имя элемента:	<code>ip</code>
Содержится в:	<code>access</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>allow</code>
Тип содержимого:	<a href="#">шаблон IP-адреса</a> (см. табл. 2.2)
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>

**Описание.** Данный элемент определяет права доступа для клиентов с заданным шаблоном IP-адреса. Если атрибут `allow` установлен в *true*, таким клиентам разрешается работа с соответствующей CGI-программой. В противном случае доступ запрещается.

### Атрибут `allow`

Имя атрибута:	<code>allow</code>
Содержится в:	<code>access</code>
Тип значения:	<i>boolean</i>
Значение по умолчанию:	<i>false</i>
Может отсутствовать:	<i>да</i>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента удовлетворяет спецификации IP-адреса в данном элементе. Если значение атрибута равно *true*, такой IP-адрес считается допустимым, а если значение атрибута установлено в *false*, доступ с этого IP-адреса запрещается.

## 2.2.4 Пример использования элемента `access`

```
<access default="deny">
  <ip allow="yes">127.</ip>
  <ip allow="no">192.168.0.14</ip>
  <ip allow="yes">192.168.0.</ip>
</access>
```

Данная спецификация ограничений IP-адресов задаёт, что клиентам, находящимся в сети класса А с адресом 127. (то есть, соединениям, устанавливаемым с того же компьютера, на котором работает WEB-сервер и CGI-программы) доступ открыт. Всем клиентам, находящимся в сети класса С с адресом 192.168.0 (то есть всем клиентам, имеющим IP-адрес в диапазоне 192.168.0.1—192.168.0.254), за исключением клиента с IP-адресом 192.168.0.14 доступ также открыт. Всем прочим клиентам доступ закрыт.

## 2.2.5 Ограничения IP-адресов в текстовом формате

Текстовый формат конфигурационного файла используется в текущей версии системы для CGI-программ `master`, `judge` и `team`. Для задания ограничений на допустимые IP-адреса используются параметры конфигурационного файла `allow_from`, `deny_from` и `allow_deny`.

Значениями параметров `allow_from` и `deny_from` является текстовая строка, в которой перечислены [спецификации IP-адресов](#) (см. табл. 2.2), разделённые пробелами. Параметр `allow_deny` принимает булевское значение. Значение *false* устанавливается по умолчанию (то есть если этот параметр не задан), либо может быть указано явно константой 0. Значение *true* устанавливается, либо если в конфигурационном файле присутствует только имя параметра без указания значения, либо при явном указании значения 1.

Параметр `allow_deny` определяет порядок просмотра IP-адресов, заданных в параметрах `allow_from` и `deny_from`.

- Если параметр `allow_deny` установлен в *false*, включается режим «недоступный по умолчанию». В этом случае IP-адрес клиента сначала сопоставляется с IP-спецификациями, перечисленными в параметре `deny_from`. Если какая-либо из IP-спецификаций сопоставляется успешно, доступ этому клиенту запрещается. Если IP-адрес не удовлетворяет ни одной из спецификаций параметра `deny_from`, IP-адрес клиента сопоставляется с IP-спецификациями, перечисленными в параметре `allow_from`. Если какая-либо из IP-спецификаций сопоставляется успешно, клиенту доступ предоставляется. Если IP-адрес не удовлетворяет ни одной из спецификаций параметра `allow_from`, доступ этому клиенту запрещается.
- Если параметр `allow_deny` установлен в *true*, включается режим «доступный по умолчанию». В этом случае IP-адрес клиента сначала сопоставляется с IP-спецификациями, перечисленными в параметре `allow_from`. Если какая-либо из IP-спецификаций сопоставляется успешно, доступ этому клиенту предоставляется. Если IP-адрес не удовлетворяет ни одной из спецификаций параметра `allow_from`, IP-адрес клиента сопоставляется с IP-спецификациями, перечисленными в параметре `deny_from`. Если какая-либо из IP-спецификаций сопоставляется успешно, доступ клиенту запрещается. Если IP-адрес не удовлетворяет ни одной из спецификаций параметра `deny_from`, доступ этому клиенту предоставляется.

В примере

```
allow_deny
```

доступ открывается клиентам со всех адресов.

В примере

```
allow_deny = 1
allow_from = "192.168.0.1 192.168.1."
deny_from = "192.168."
```

доступ открывается клиентом со всех адресов, IP-адреса которых не начинаются с 192.168. Из клиентов с IP-адресами, начинающимися с 192.168 допускаются лишь клиенты с IP-адресом, начинающимся на 192.168.1 и клиенты с IP-адресом 192.168.0.1.

В примере

```
allow_deny = 0
allow_from = "127. 192.168.0.1 192.168.1."
```

доступ закрывается всем, кроме клиентов с IP-адресом, начинающихся с 127 (то есть для локально устанавливаемых соединений), клиентом с IP-адресом, начинающимся с 192.168.1 и клиентов с IP-адресом 192.168.0.1.

## 2.3 Локализация системы

Система **ejudge** поддерживает настройку языка пользовательского интерфейса, хотя текущая версия системы обладает рядом недостатков.

Переключения языка пользовательского интерфейса выполняется через стандартный механизм трансляции сообщений `gettext`. В настоящее время система поддерживает только два языковых окружения — `C` (окружение по умолчанию для программ на Си) и `ru_RU.KOI8-R` (русский язык в кодировке `koi-r`), которые имеют идентификаторы 0 и 1 соответственно. Для того, чтобы добавить новое языковое окружение, необходимо модифицировать исходный файл `l10n.c`. Переведённые тексты сообщений находятся в файлах `ejudge.LOCALE.po`, где *LOCALE* — имя языкового окружения.

### 2.3.1 Включение поддержки языкового окружения

Чтобы включить поддержку языкового окружения, необходимо установить конфигурационный параметр в файлах настройки утилит. Для конфигурационных файлов в формате XML ([ejudge.xml](#), [register.xml](#), [users.xml](#)) это атрибут `l10n` элемента верхнего уровня конфигурационного файла. Например, для конфигурационного файла `ejudge.xml` установка атрибута локализации сообщений может выглядеть следующим образом:

```
<?xml version="1.0" encoding="koi8-r"?>
<config l10n="yes">
  <!-- прочие элементы -->
</config>
```

Для конфигурационных файлов в текстовом формате ([serve.cfg](#), [master.cfg](#), [judge.cfg](#), [team.cfg](#)) это параметр `enable_l10n`, который необходимо установить. Для любого конфигурационного установка атрибута локализации может выглядеть следующим образом:

```
enable_l10n
```



### 2.3.2 Установка каталога сообщений

Чтобы система **ejudge** смогла использовать переводы сообщений, в конфигурационных файлах утилит необходимо указать каталог сообщений (message catalog), в котором находятся файлы с переведёнными сообщениями. По умолчанию после выполнения команды `make install` каталоги сообщений помещаются в каталог, определяемой переменной `INST_LOCALE_PATH` в `makefile`. Именно этот каталог должен быть указан в конфигурационных файлах. Для конфигурационных файлов в формате XML (`ejudge.xml`, `register.xml`, `users.xml`) каталог сообщений устанавливается с помощью элемента `l10n_dir`, например, следующим образом:

```
<l10n_dir>/usr/share/locale</l10n_dir>
```

Для конфигурационных файлов в текстовом формате (`serve.cfg`, `master.cfg`, `judge.cfg`, `team.cfg`) каталог сообщений устанавливается с помощью параметра `l10n_dir`, например, следующим образом:

```
l10n_dir = "/usr/share/locale"
```

### 2.3.3 Поддержка кодировок символов

Поддержка кодировок для ввода/вывода реализована в текущей версии системы недостаточным образом. Внутренней кодировкой для системы **ejudge** в настоящее время является кодировка `koi8-r`. Все текстовые строки представляются в памяти именно в этой кодировке. Это значит, что текстовые строки в других кодировках окажутся искажены. Корректная реализация системы должны хранить строки в кодировке `utf-8` или `utf-16`. Кроме того, пользователь должен иметь возможность задавать желательную кодировку символов.

#### Кодировка конфигурационных файлов

Кодировка конфигурационных файлов в формате XML (`ejudge.xml`, `register.xml`, `users.xml`, `contest.xml`, `userlist.xml`) задаётся непосредственно в заголовке соответствующего файла, например,

```
<?xml version="1.0" encoding="koi8-r"?>
```

Текущая версия системы поддерживает следующие кодировки XML-файлов: `iso8859-1`, `utf-8`, `utf-16`, `koi8-r`, `cp1251`, `cp866`, `iso8859-5`. Однако при чтении файла текст в любой кодировке конвертируется в текст в кодировке `koi8-r`.

Кодировка конфигурационных файлов в текстовом формате (`serve.cfg`, `master.cfg`, `judge.cfg`, `team.cfg`) в настоящее время не может быть задана явно, и всегда подразумевается кодировка `koi8-r`.

#### Кодировка генерируемых файлов

При генерации `html`-страниц используется кодировка, заданная в конфигурационном файле соответствующей утилиты. Если параметр кодировки не задан, используется кодировка `iso8859-1`. В текущей версии системы **ejudge** параметр кодировки вывода используется только для корректной генерации `http`-заголовка. Не делается попытка перекодировки символов, если кодировка какой-либо части генерируемой `html`-страницы (например, имён пользователей) не соответствует указанной кодировке. Это практически наверняка приведёт к некорректному отображению страницы, если указана кодировка вывода, отличная от `iso8859-1` или `koi8-r`.



Исключением в этой ситуации является таблица текущих результатов турнира. Кодировка этой html-страницы может быть задана явно с помощью параметра `standings_charset` конфигурационного файла `serve.cfg`. Если эта кодировка отличается от кодировки `koi8-r`, будет выполнено необходимое преобразование.

Для конфигурационных файлов утилит в формате XML (`register.xml`, `users.xml`) кодировка генерируемой страницы указывается с помощью атрибута `charset` элемента верхнего уровня. Например, в конфигурационном файле `register.xml` кодировка может быть задана следующим образом:

```
<?xml version="1.0" encoding="koi8-r"?>
<register_config
  charset = "KOI8-R"
  <!-- прочие атрибуты -->
>
  <!-- прочие элементы -->
</register_config>
```

Для конфигурационных файлов утилит в текстовом формате (`serve.cfg`, `master.cfg`, `judge.cfg`, `team.cfg`) кодировка генерируемых html-страниц указывается с помощью параметра `charset`, например, следующим образом: `verbatim`

```
charset = "koi8-r"
```

## 2.4 Общий конфигурационный файл `ejudge.xml`

Данный конфигурационный файл, который далее в тексте этого документа будет называться `ejudge.xml` содержит настройки утилит командной строки, как непосредственно работающих с базой данных, так и являющихся клиентами сервера пользователей. Файл используется утилитами `clean-users`, `edit-userlist`, `super-serve`, `userlist-server`.

## 2.4.1 Общая структура

Конфигурационный файл должен быть корректным XML-файлом. Он должен состоять из единственного элемента первого уровня `config`. Иерархия элементов приведена на схеме ниже.

```
config
  caps
    cap
  contests_dir
  email_program
  l10n_dir
  register_email
  register_url
  serve_path
  run_path
  socket_path
  userdb_file
  user_map
    map
```

Описание каждого элемента дано ниже.

## 2.4.2 Тег `config`

Имя элемента:	<b>config</b>
Содержится в:	<i>нет</i>
Может содержать:	<code>caps</code> , <code>contests_dir</code> , <code>email_program</code> , <code>l10n_dir</code> , <code>register_email</code> , <code>register_url</code> , <code>serve_path</code> , <code>run_path</code> , <code>socket_path</code> , <code>userdb_file</code> , <code>user_map</code>
Атрибуты:	<code>l10n</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>clean-users</code> , <code>edit-userlist</code> , <code>super-serve</code> , <code>userlist-server</code>
Описание.	Тег верхнего уровня конфигурационного файла.

### 2.4.3 Тег cap

Имя элемента:	<b>cap</b>
Содержится в:	<a href="#">caps</a>
Может содержать:	<i>нет</i>
Атрибуты:	<a href="#">login</a>
Тип содержимого:	<a href="#">список битов полномочий</a>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>
Используется в:	<a href="#">userlist-server</a>

**Описание.** Данный элемент позволяет задавать полномочия на выполнение операций над базой данных пользователей. Описание полномочий, поддерживаемых системой ejudge, приведено в разделе [2.1](#).

Атрибут `login` задаёт регистрационное имя пользователя, для которого устанавливаются полномочия.

**Пример.**

```
<cap login="dima">MAP_CONTEST</cap>
```

### 2.4.4 Тег caps

Имя элемента:	<b>caps</b>
Содержится в:	<a href="#">config</a>
Может содержать:	<a href="#">cap</a>
Атрибуты:	<i>нет</i>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>
Используется в:	<a href="#">userlist-server</a>

**Описание.** Данный элемент является элементом верхнего уровня списка списков полномочий пользователей по отношению к базе данных пользователей. Элемент внутри себя может содержать только элементы `cap`. Если элемент `caps` встречается в конфигурационном файле несколько раз, списки списков полномочий сливаются друг с другом.

**Пример.**

```
<caps>
  <cap login="a"></cap>
  <cap login="b"></cap>
</caps>
```

## 2.4.5 Тег `contests_dir`

Имя элемента:	<code>contests_dir</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>clean-users</code> , <code>edit-userlist</code> , <code>super-serve</code> , <code>userlist-server</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
<contests_dir>/var/ejudge/contests</contests_dir>
```

## 2.4.6 Тег `email_program`

Имя элемента:	<code>email_program</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к программе
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент задаёт путь программе отсылки электронной почты. Эта программа запускается без аргументов командной строки, и на её стандартный поток ввода подаются служебные поля письма (`From`, `To` и т. д.), а затем текст письма. Почтовое сообщение отсылается по адресу, указанному в регистрационной анкете, при регистрации нового пользователя и содержит его регистрационное имя (`login`) и пароль.

**Пример.**

```
<email_program>/var/qmail/bin/qmail-inject</email_program>
```

## 2.4.7 Тег `l10n_dir`

Имя элемента:	<code>l10n_dir</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся локализованные сообщения. Для локализации сообщений используется GNU `gettext`. См. раздел ??? для информации о локализации системы.

**Пример.**

```
<l10n_dir>/usr/share/ejudge/locale</l10n_dir>
```

## 2.4.8 Тег `map`

Имя элемента:	<code>map</code>
Содержится в:	<code>user_map</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>system_user</code> <code>local_user</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент задаёт отображение регистрационных имён UNIX в регистрационные имена системы `ejudge`. Атрибут `system_user` задаёт регистрационное имя UNIX. Пользователь с таким регистрационным именем должен существовать в системе. Атрибут `local_user` задаёт регистрационное имя системы `ejudge`.

**Пример.**

```
<map system_user="aaa" local_user="tolik"/>
```

## 2.4.9 Тег `register_email`

Имя элемента:	<code>register_email</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	адрес электронной почты
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент задаёт электронный адрес, от имени которого рассылаются уведомления о регистрации новых пользователей. Если этот элемент неопределён, используется адрес по умолчанию `team@contest.cmc.msu.ru`. Значение, устанавливаемое этим элементом может быть переопределено элементом `register_email` конфигурационного файла турнира `contest.xml`.

**Пример.**

```
<register_email>ejudge@olympiads.ru</register_email>
```

## 2.4.10 Тег `register_url`

Имя элемента:	<code>register_url</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	URL
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент задаёт URL CGI программы `register`, которая отвечает за регистрацию новых пользователей, изменение пользователями своих персональных данных и регистрацию пользователей на турниры. Этот URL добавляется в текст уведомления о регистрации. Если этот элемент неопределён, используется URL по умолчанию `http://contest.cmc.msu.ru/cgi-bin/register`. Значение, устанавливаемое этим элементом может быть переопределено элементом `register_url` конфигурационного файла турнира `contest.xml`.

**Пример.**

```
<register_url>http://www.olympiads.ru/cgi-bin/register</register_url>
```

## 2.4.11 Тег `serve_path`

Имя элемента:	<code>serve_path</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к программе <code>serve</code>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>super-serve</code>
Описание.	Данный элемент задаёт путь к программе <code>serve</code> системы ejudge.
Пример.	

```
<serve_path>/usr/ejudge/bin/serve</serve_path>
```

## 2.4.12 Тег `run_path`

Имя элемента:	<code>run_path</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к программе <code>run</code>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>super-serve</code>
Описание.	Данный элемент задаёт путь к программе <code>run</code> системы ejudge.
Пример.	

```
<serve_path>/usr/ejudge/bin/run</serve_path>
```

## 2.4.13 Тег `socket_path`

Имя элемента:	<code>socket_path</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к UNIX-сокету
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>edit-userlist</code> , <code>userlist-server</code>

**Описание.** Данный элемент задаёт путь к UNIX-сокету, который используется для связи с сервером информации о пользователях `userlist-server`. Сокет создаётся при старте этой программы, поэтому должен находиться в каталоге, доступном ей на запись. Смотри также элемент `socket_path` конфигурационного файла `users.xml`, элемент `socket_path` конфигурационного файла `register.xml`, параметр `socket_path` конфигурационного файла `master.cfg`, параметр `socket_path` конфигурационного файла `judge.cfg`, параметр `socket_path` конфигурационного файла `team.cfg`, параметр `socket_path` конфигурационного файла `serve.cfg`.

**Пример.**

```
<socket_path>/var/ejudge/userlist-socket</socket_path>
```



## 2.4.14 Тег `user_map`

Имя элемента:	<code>user_map</code>
Содержится в:	<code>config</code>
Может содержать:	<code>map</code>
Атрибуты:	<i>нет</i>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>userlist-server</code>

**Описание.** Данный элемент является элементом верхнего уровня списка отображения системных (Unix) пользователей в пользователей ejudge. Используется для определения уровня привилегий пользователя при запуске им утилит `edit-userlist` и `serve`, когда последние подключаются к базе данных пользователей.

**Пример.**

```
<user_map>
  <map system_user="john" local_user="vasya_petrov"/>
</user_map>
```

## 2.4.15 Тег `userdb_file`

Имя элемента:	<code>userdb_file</code>
Содержится в:	<code>config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Используется в:	<code>clean-users</code> , <code>userlist-server</code>

**Описание.** Текст в данном элементе задаёт путь к файлу, хранящему информацию о пользователях системы `userlist.xml`.

**Пример.**

```
<userdb_file>/var/ejudge/db/userlist.xml</userdb_file>
```

## 2.5 Конфигурационный файл `register.xml`

Этот конфигурационный файл используется CGI-программой `register` для начальной инициализации после старта. По умолчанию конфигурационный файл должен находиться в каталоге `../cgi-data` относительно каталога, в котором находится исполняемый файл `register`. Изменение конфигурационного каталога возможно при компиляции программы в `makefile`. Если программа `register` запускается под другим именем, то имя конфигурационного файла будет соответствующим образом изменяться. Точные правила поиска конфигурационного файла приведены в разделе ???.

Например, если программа запускается под именем `register-N`, где  $N$  — некоторое десятичное число, то сначала будет сделана попытка считать конфигурационный файл `../cgi-data/register-N.xml`, затем, если эта попытка завершилась неудачно, будет сделана попытка считать конфигурационный файл `../cgi-data/register-N'.xml`, где  $N'$  — это число, напечатанное с 6 десятичными знаками, включая незначащие нули, например, `../cgi-data/register-000001.xml`. Если эта попытка завершилась неудачно, будет сделана попытка считать конфигурационный файл `../cgi-data/register-N''.xml`, где  $N''$  — число, напечатанное без ведущих незначащих нулей. Наконец, будет сделана попытка считать конфигурационный файл `../cgi-data/register.xml`, и в случае неуспеха программа сгенерирует сообщение об ошибке и завершится.

Конфигурационный файл представляет собой правильный XML-файл, элементом верхнего уровня которого является `register_config`.

Общая структура файла приведена ниже.

```
register_config
  access
  ip
  contests_dir
  l10n_dir
  socket_path
```

Далее даётся описание всех элементов конфигурационного файла.

### 2.5.1 Тег `register_config`

Имя элемента:	<code>register_config</code>
Содержится в:	<i>нет</i>
Может содержать:	<code>access</code> , <code>contests_dir</code> , <code>l10n_dir</code> , <code>socket_path</code>
Атрибуты:	<code>l10n</code> , <code>charset</code> , <code>show_generation_time</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Описание.	Тег верхнего уровня конфигурационного файла.

## Атрибут `l10n`

Имя атрибута:	<code>l10n</code>
Содержится в:	<code>register_config</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>da</code>

**Описание.** Данный атрибут включает поддержку локализации интерфейса. Если его значение установлено в `true`, CGI-программа `register` считывает значение параметра `locale_id` и переключает язык интерфейса в зависимости от значения параметра. Подробнее вопросы локализации системы рассматриваются в разделе ???.

Логическое значение `true` задаётся с помощью строк `1`, `yes` или `true`, а логическое значение `false` — с помощью строк `0`, `no`, `false`.

## Атрибут `charset`

Имя атрибута:	<code>charset</code>
Содержится в:	<code>register_config</code>
Тип значения:	<code>string</code>
Значение по умолчанию:	<code>iso8859-1</code>
Может отсутствовать:	<code>da</code>

**Описание.** Данный атрибут позволяет задавать кодировку, которая будет проставляться в заголовке всех генерируемых страниц. Обратите внимание, что в текущей версии кодировка проставляется вне зависимости от языка интерфейса.

## Атрибут `show_generation_time`

Имя атрибута:	<code>show_generation_time</code>
Содержится в:	<code>register_config</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>da</code>

**Описание.** Если данный атрибут установлен в `true`, генерируемые html-страницы будут содержать информацию о процессорном времени в миллисекундах, затраченном на генерацию каждой страницы. Например, если язык интерфейса — английский, сообщение будет иметь примерно следующий вид:

```
Page generation time: 37 msec
```

## 2.5.2 Тег `access`

Имя элемента:	<b>access</b>
Содержится в:	<code>register_config</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задавать ограничения на IP-адреса, с которых производится обращение к CGI-программе `register`. Ограничения, заданные в этом конфигурационном файле, проверяются до вывода регистрационной страницы и действуют на все турниры, определённые в системе. Эти ограничения могут использоваться для запрещения любого доступа к системе от «нежелательных» IP-адресов. Каждый турнир может накладывать дополнительные ограничения. Они задаются с помощью элемента `register_access` конфигурационного файла `contest.xml`. См. подробное описание ограничений IP-адреса в разделе 2.2.

### Атрибут `default`

Имя атрибута:	<b>default</b>
Содержится в:	<code>access</code>
Тип значения:	<i>allow</i> или <i>deny</i>
Значение по умолчанию:	<i>deny</i>
Может отсутствовать:	<i>да</i>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента не перечислен в списке IP-адресов. Если атрибут установлен в *allow*, такие IP-адреса считаются допустимыми, а если атрибут установлен в *deny*, доступ с таких IP-адресов запрещается. См. подробное описание ограничений IP-адреса в разделе 2.2.

## 2.5.3 Тег `ip`

Имя элемента:	<b>ip</b>
Содержится в:	<code>access</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>allow</code>
Тип содержимого:	шаблон IP-адреса
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>

**Описание.** Данный элемент определяет права доступа для клиентов с заданным шаблоном IP-адреса. Если атрибут `allow` установлен в *true*, таким клиентам разрешается работа с CGI-программой `register`. См. подробное описание ограничений IP-адреса в разделе 2.2.

## Атрибут `allow`

Имя атрибута:	<code>allow</code>
Содержится в:	<code>access</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>да</code>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента удовлетворяет спецификации IP-адреса в данном элементе. Если значение атрибута равно `true`, такой IP-адрес считается допустимым, а если значение атрибута установлено в `false`, доступ с этого IP-адреса запрещается. См. подробное описание ограничений IP-адреса в разделе [2.2](#).

### 2.5.4 Тег `l10n_dir`

Имя элемента:	<code>l10n_dir</code>
Содержится в:	<code>register_config</code>
Может содержать:	<code>нет</code>
Атрибуты:	<code>нет</code>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся локализованные сообщения. Для локализации сообщений используется GNU `gettext`. См. раздел ??? для информации о локализации системы.

**Пример.**

```
<l10n_dir>/usr/share/ejudge/locale</l10n_dir>
```

### 2.5.5 Тег `contests_dir`

Имя элемента:	<code>contests_dir</code>
Содержится в:	<code>register_config</code>
Может содержать:	<code>нет</code>
Атрибуты:	<code>нет</code>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
<contests_dir>/var/ejudge/contests</contests_dir>
```

## 2.5.6 Тег `socket_path`

<b>Имя элемента:</b>	<b><code>socket_path</code></b>
<b>Содержится в:</b>	<code>register_config</code>
<b>Может содержать:</b>	<i>нет</i>
<b>Атрибуты:</b>	<i>нет</i>
<b>Тип содержимого:</b>	путь к UNIX-сокету
<b>Может отсутствовать:</b>	<i>нет</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данный элемент задаёт путь к UNIX-сокету, который используется для связи с сервером информации о пользователях `userlist-server`. Когда программа `register` стартует, UNIX-сокет должен существовать и быть доступным для пользователя, от имени которого запускается программа. Как правило, это пользователь, от имени которого работает WEB-сервер.

**Пример.**

```
<socket_path>/var/ejudge/userlist-socket</socket_path>
```

## 2.6 Конфигурационный файл `users.xml`

Этот конфигурационный файл используется CGI-программой `users` для начальной инициализации после старта. По умолчанию конфигурационный файл должен находиться в каталоге `../cgi-data` относительно каталога, в котором находится исполняемый файл `users`. Изменение конфигурационного каталога возможно при компиляции программы в `makefile`. Если программа `users` запускается под другим именем, то имя конфигурационного файла будет соответствующим образом изменяться. Точные правила поиска конфигурационного файла приведены в разделе ???.

Например, если программа запускается под именем `users-N`, где  $N$  — некоторое десятичное число, то сначала будет сделана попытка считать конфигурационный файл `../cgi-data/users-N.xml`, затем, если эта попытка завершилась неудачно, будет сделана попытка считать конфигурационный файл `../cgi-data/users-N'.xml`, где  $N'$  — это число, напечатанное с 6 десятичными знаками, включая незначащие нули, например, `../cgi-data/users-000001.xml`. Если это попытка завершилась неудачно, будет сделана попытка считать конфигурационный файл `../cgi-data/users-N''.xml`, где  $N''$  — число, напечатанное без ведущих незначащих нулей. Наконец, будет сделана попытка считать конфигурационный файл `../cgi-data/users.xml`, и в случае неуспеха программа сгенерирует сообщение об ошибке и завершится.

Конфигурационный файл представляет собой правильный XML-файл, элементом верхнего уровня которого является `users_config`.

Общая структура файла приведена ниже.

```
users_config
  access
    ip
  contests_dir
  l10n_dir
  socket_path
```

Далее даётся описание всех элементов конфигурационного файла.

### 2.6.1 Тег `users_config`

Имя элемента:	<code>users_config</code>
Содержится в:	<i>нет</i>
Может содержать:	<code>access</code> , <code>contests_dir</code> , <code>l10n_dir</code> , <code>socket_path</code>
Атрибуты:	<code>l10n</code> , <code>charset</code> , <code>show_generation_time</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Описание.	Тег верхнего уровня конфигурационного файла.

## Атрибут 110n

Имя атрибута:	<b>110n</b>
Содержится в:	<code>users_config</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>da</code>

**Описание.** Данный атрибут включает поддержку локализации интерфейса. Если его значение установлено в `true`, CGI-программа `users` считывает значение параметра `locale_id` и переключает язык интерфейса в зависимости от значения параметра. Подробнее вопросы локализации системы рассматриваются в разделе ???.

Логическое значение `true` задаётся с помощью строк 1, yes или true, а логическое значение `false` — с помощью строк 0, no, false.

## Атрибут charset

Имя атрибута:	<b>charset</b>
Содержится в:	<code>users_config</code>
Тип значения:	<code>string</code>
Значение по умолчанию:	<code>iso8859-1</code>
Может отсутствовать:	<code>da</code>

**Описание.** Данный атрибут позволяет задавать кодировку, которая будет проставляться в заголовке всех генерируемых страниц. Обратите внимание, что в текущей версии кодировка проставляется вне зависимости от языка интерфейса.

## Атрибут show\_generation\_time

Имя атрибута:	<b>show_generation_time</b>
Содержится в:	<code>users_config</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>da</code>

**Описание.** Если данный атрибут установлен в `true`, генерируемые html-страницы будут содержать информацию о процессорном времени в миллисекундах, затраченном на генерацию каждой страницы. Например, если язык интерфейса — английский, сообщение будет иметь примерно следующий вид:

```
Page generation time: 37 msec
```



## 2.6.2 Тег `access`

Имя элемента:	<b>access</b>
Содержится в:	<code>users_config</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задавать ограничения на IP-адреса, с которых производится обращение к CGI-программе `users`. Ограничения, заданные в этом конфигурационном файле, проверяются до вывода регистрационной страницы и действуют на все турниры, определённые в системе. Эти ограничения могут использоваться для запрещения любого доступа к системе от «нежелательных» IP-адресов. Каждый турнир может накладывать дополнительные ограничения. Они задаются с помощью элемента `users_access` конфигурационного файла `contest.xml`. См. подробное описание ограничений IP-адреса в разделе 2.2.

### Атрибут `default`

Имя атрибута:	<b>default</b>
Содержится в:	<code>access</code>
Тип значения:	<i>allow</i> или <i>deny</i>
Значение по умолчанию:	<i>deny</i>
Может отсутствовать:	<i>да</i>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента не перечислен в списке IP-адресов. Если атрибут установлен в *allow*, такие IP-адреса считаются допустимыми, а если атрибут установлен в *deny*, доступ с таких IP-адресов запрещается. См. подробное описание ограничений IP-адреса в разделе 2.2.

## 2.6.3 Тег `ip`

Имя элемента:	<b>ip</b>
Содержится в:	<code>access</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>allow</code>
Тип содержимого:	шаблон IP-адреса
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>

**Описание.** Данный элемент определяет права доступа для клиентов с заданным шаблоном IP-адреса. Если атрибут `allow` установлен в *true*, таким клиентам разрешается работа с CGI-программой `users`. См. подробное описание ограничений IP-адреса в разделе 2.2.

## Атрибут `allow`

Имя атрибута:	<code>allow</code>
Содержится в:	<code>access</code>
Тип значения:	<code>boolean</code>
Значение по умолчанию:	<code>false</code>
Может отсутствовать:	<code>да</code>

**Описание.** Данный атрибут задаёт права доступа в случае, если IP-адрес клиента удовлетворяет спецификации IP-адреса в данном элементе. Если значение атрибута равно `true`, такой IP-адрес считается допустимым, а если значение атрибута установлено в `false`, доступ с этого IP-адреса запрещается. См. подробное описание ограничений IP-адреса в разделе [2.2](#).

### 2.6.4 Тег `l10n_dir`

Имя элемента:	<code>l10n_dir</code>
Содержится в:	<code>users_config</code>
Может содержать:	<code>нет</code>
Атрибуты:	<code>нет</code>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся локализованные сообщения. Для локализации сообщений используется GNU `gettext`. См. раздел ??? для информации о локализации системы.

**Пример.**

```
<l10n_dir>/usr/share/ejudge/locale</l10n_dir>
```

### 2.6.5 Тег `contests_dir`

Имя элемента:	<code>contests_dir</code>
Содержится в:	<code>users_config</code>
Может содержать:	<code>нет</code>
Атрибуты:	<code>нет</code>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
<contests_dir>/var/ejudge/contests</contests_dir>
```

## 2.6.6 Тег `socket_path`

Имя элемента:	<code>socket_path</code>
Содержится в:	<code>users_config</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к UNIX-сокету
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент задаёт путь к UNIX-сокету, который используется для связи с сервером информации о пользователях `userlist-server`. Когда программа `users` стартует, UNIX-сокет должен существовать и быть доступным для пользователя, от имени которого запускается программа. Как правило, это пользователь, от имени которого работает WEB-сервер.

**Пример.**

```
<socket_path>/var/ejudge/userlist-socket</socket_path>
```

## 2.7 Конфигурационный файл `master.cfg`

Данный конфигурационный файл используется CGI-утилитой `master`. По умолчанию он должен находиться в каталоге `../cgi-data` относительно каталога, из которого запускается программа `master`. Изменение каталога по умолчанию возможно при компиляции системы `ejudge` изменением соответствующей переменной настройки в `makefile`. При запуске программы `master` она пытается считать конфигурационный файл из конфигурационного файла, причём имя конфигурационного файла может зависеть от номера турнира и имени, под которым запускается программа `master`. Точные правила поиска конфигурационного файла даны в разделе ???. Конфигурационный файл, считываемый, если никакой другой конфигурационный файл считать не удалось, называется `master.cfg`. Если и чтение этого файла завершилось с ошибкой, программа `master` генерирует сообщение ошибки конфигурации системы. Далее любой конфигурационный файл, используемый программой `master` будет называться `master.cfg`.

Конфигурационный файл `master.cfg` имеет текстовый формат, напоминающий формат `.ini`-файлов. Каждый конфигурационный параметр задаётся в виде

```
<имя параметра> = <значение>
```

где `<имя параметра>` может состоять из латинских заглавных и строчных букв, цифр и знака подчёркивания. `<значение>` должно заключаться в кавычки, если оно содержит пробельные символы, но может быть задано и без кавычек, если оно состоит из единственного слова. Пробельные символы в начале и конце строки и вокруг знака `-` игнорируются. Комментарии в конфигурационном файле начинаются с символов `#` или `;` и продолжаются до конца строки.

Параметры конфигурационного файла перечислены ниже.

## 2.7.1 Параметр `allow_deny`

Имя параметра:	<code>allow_deny</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.7.3</a>

## 2.7.2 Параметр `allow_from`

Имя параметра:	<code>allow_from</code>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.7.3</a>

## 2.7.3 Параметр `deny_from`

Имя параметра:	<code>deny_from</code>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример.	

```
allow_deny = 1
allow_from = "192.168.0.1 192.168.1."
deny_from = "192.168."
```

## 2.7.4 Параметр `charset`

Имя параметра:	<code>charset</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>iso8859-1</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт название кодировки, которое будет помещаться в http-заголовок генерируемой страницы. Полная информация о локализации системы **ejudge** приведена в разделе [2.3.3](#).

**Пример.**

```
charset = "koi8-r"
```

## 2.7.5 Параметр `enable_l10n`

Имя параметра:	<b><code>enable_l10n</code></b>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>
Пример:	См. раздел <a href="#">2.7.6</a>
Замечание:	В текущей версии программы <b>master</b> данный параметр не имеет никакого эффекта.

## 2.7.6 Параметр `l10n_dir`

Имя параметра:	<b><code>l10n_dir</code></b>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>
Замечание:	В текущей версии программы <b>master</b> данный параметр не имеет никакого эффекта.

Если параметр `enable_l10n` установлен в *true*, а параметр `l10n_dir` не задан, локализация сообщений принудительно отключается.

**Пример.**

```
enable_l10n
l10n_dir = "/usr/share/locale"
```

## 2.7.7 Параметр `socket_path`

Имя параметра:	<b><code>socket_path</code></b>
Тип содержимого:	путь к UNIX-сокету
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт путь к UNIX-сокету, который используется для связи с программой-сервером базы пользователей [userlist-server](#). В момент запуска программы **master** сокет должен существовать.

**Пример.**

```
<socket_path>/var/ejudge/userlist-socket</socket_path>
```

## 2.7.8 Параметр `contests_dir`

**Имя параметра:** `contests_dir`

**Тип содержимого:** путь к каталогу

**Может отсутствовать:** *нет*

**Может повторяться:** *нет*

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
contests_dir = "/var/ejudge/contests"
```

## 2.8 Конфигурационный файл `judge.cfg`

Данный конфигурационный файл используется CGI-утилитой `judge`. По умолчанию он должен находиться в каталоге `../cgi-data` относительно каталога, из которого запускается программа `judge`. Изменение каталога по умолчанию возможно при компиляции системы `ejudge` изменением соответствующей переменной настройки в `makefile`. При запуске программы `judge` она пытается считать конфигурационный файл из конфигурационного файла, причём имя конфигурационного файла может зависеть от номера турнира и имени, под которым запускается программа `judge`. Точные правила поиска конфигурационного файла даны в разделе ???. Конфигурационный файл, считываемый, если никакой другой конфигурационный файл считать не удалось, называется `judge.cfg`. Если и чтение этого файла завершилось с ошибкой, программа `judge` генерирует сообщение ошибки конфигурации системы. Далее любой конфигурационный файл, используемый программой `judge` будет называться `judge.cfg`.

Формат конфигурационного файла `judge.cfg` в точности совпадает с форматом конфигурационного файла `master.cfg`, так как программы `master` и `judge` — это одна программа, запускаемая под разными именами. Конфигурационный файл `judge.cfg` имеет текстовый формат, напоминающий формат `.ini`-файлов. Каждый конфигурационный параметр задаётся в виде

```
<имя параметра> = <значение>
```

где `<имя параметра>` может состоять из латинских заглавных и строчных букв, цифр и знака подчёркивания. `<значение>` должно заключаться в кавычки, если оно содержит пробельные символы, но может быть задано и без кавычек, если оно состоит из единственного слова. Пробельные символы в начале и конце строки и вокруг знака `"` игнорируются. Комментарии в конфигурационном файле начинаются с символов `#` или `;` и продолжаются до конца строки.

Параметры конфигурационного файла перечислены ниже.

## 2.8.1 Параметр `allow_deny`

Имя параметра:	<b><code>allow_deny</code></b>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.8.3</a>

## 2.8.2 Параметр `allow_from`

Имя параметра:	<b><code>allow_from</code></b>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.8.3</a>

## 2.8.3 Параметр `deny_from`

Имя параметра:	<b><code>deny_from</code></b>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример.	

```
allow_deny = 1
allow_from = "192.168.0.1 192.168.1."
deny_from = "192.168."
```

## 2.8.4 Параметр `charset`

Имя параметра:	<b><code>charset</code></b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>iso8859-1</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт название кодировки, которое будет помещаться в http-заголовок генерируемой страницы. Полная информация о локализации системы **ejudge** приведена в разделе [2.3.3](#).

**Пример.**

```
charset = "koi8-r"
```

## 2.8.5 Параметр `enable_l10n`

Имя параметра:	<b><code>enable_l10n</code></b>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>
Пример:	См. раздел <a href="#">2.8.6</a>
Замечание:	В текущей версии программы <code>judge</code> данный параметр не имеет никакого эффекта.

## 2.8.6 Параметр `l10n_dir`

Имя параметра:	<b><code>l10n_dir</code></b>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>
Замечание:	В текущей версии программы <code>judge</code> данный параметр не имеет никакого эффекта.

Если параметр `enable_l10n` установлен в *true*, а параметр `l10n_dir` не задан, локализация сообщений принудительно отключается.

**Пример.**

```
enable_l10n
l10n_dir = "/usr/share/locale"
```

## 2.8.7 Параметр `socket_path`

Имя параметра:	<b><code>socket_path</code></b>
Тип содержимого:	путь к UNIX-сокету
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт путь к UNIX-сокету, который используется для связи с программой-сервером базы пользователей [userlist-server](#). В момент запуска программы `judge` сокет должен существовать.

**Пример.**

```
socket_path = "/var/ejudge/userlist-socket"
```



## 2.8.8 Параметр `contests_dir`

**Имя параметра:** `contests_dir`

**Тип содержимого:** путь к каталогу

**Может отсутствовать:** *нет*

**Может повторяться:** *нет*

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
contests_dir = "/var/ejudge/contests"
```

## 2.9 Конфигурационный файл `team.cfg`

Данный конфигурационный файл используется CGI-утилитой `team`. По умолчанию он должен находиться в каталоге `../cgi-data` относительно каталога, из которого запускается программа `team`. Изменение каталога по умолчанию возможно при компиляции системы `ejudge` изменением соответствующей переменной настройки в `makefile`. При запуске программы `team` она пытается считать конфигурационный файл из конфигурационного файла, причём имя конфигурационного файла может зависеть от номера турнира и имени, под которым запускается программа `team`. Точные правила поиска конфигурационного файла даны в разделе ????. Конфигурационный файл, считываемый, если никакой другой конфигурационный файл считать не удалось, называется `team.cfg`. Если и чтение этого файла завершилось с ошибкой, программа `team` генерирует сообщение ошибки конфигурации системы. Далее любой конфигурационный файл, используемый программой `team` будет называться `team.cfg`.

Формат конфигурационного файла `team.cfg` практически совпадает с форматом конфигурационного файла `master.cfg`. Конфигурационный файл `team.cfg` имеет текстовый формат, напоминающий формат `.ini`-файлов. Каждый конфигурационный параметр задаётся в виде

```
<имя параметра> = <значение>
```

где `<имя параметра>` может состоять из латинских заглавных и строчных букв, цифр и знака подчёркивания. `<значение>` должно заключаться в кавычки, если оно содержит пробельные символы, но может быть задано и без кавычек, если оно состоит из единственного слова. Пробельные символы в начале и конце строки и вокруг знака `-` игнорируются. Комментарии в конфигурационном файле начинаются с символов `#` или `;` и продолжаются до конца строки.

Параметры конфигурационного файла перечислены ниже.

## 2.9.1 Параметр `allow_deny`

Имя параметра:	<b><code>allow_deny</code></b>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.9.3</a>

## 2.9.2 Параметр `allow_from`

Имя параметра:	<b><code>allow_from</code></b>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример:	См. раздел <a href="#">2.9.3</a>

## 2.9.3 Параметр `deny_from`

Имя параметра:	<b><code>deny_from</code></b>
Тип содержимого:	<i>спецификация IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.2.5</a>
Пример.	

```
allow_deny = 1
allow_from = "192.168.0.1 192.168.1."
deny_from = "192.168."
```

## 2.9.4 Параметр `charset`

Имя параметра:	<b><code>charset</code></b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>iso8859-1</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт название кодировки, которое будет помещаться в http-заголовок генерируемой страницы. Полная информация о локализации системы **ejudge** приведена в разделе [2.3.3](#).

**Пример.**

```
charset = "koi8-r"
```

## 2.9.5 Параметр `enable_l10n`

Имя параметра:	<code>enable_l10n</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>
Пример:	См. раздел <a href="#">2.9.6</a>

## 2.9.6 Параметр `l10n_dir`

Имя параметра:	<code>l10n_dir</code>
Тип содержимого:	путь к каталогу
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	пустая строка
Может повторяться:	<i>нет</i>
Описание:	См. раздел <a href="#">2.3.1</a>

Если параметр `enable_l10n` установлен в *true*, а параметр `l10n_dir` не задан, локализация сообщений принудительно отключается.

**Пример.**

```
enable_l10n
l10n_dir = "/usr/share/locale"
```

## 2.9.7 Параметр `socket_path`

Имя параметра:	<code>socket_path</code>
Тип содержимого:	путь к UNIX-сокету
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный параметр задаёт путь к UNIX-сокету, который используется для связи с программой-сервером базы пользователей [userlist-server](#). В момент запуска программы `team` сокет должен существовать.

**Пример.**

```
socket_path = "/var/ejudge/userlist-socket"
```

## 2.9.8 Параметр `contests_dir`

**Имя параметра:** `contests_dir`

**Тип содержимого:** путь к каталогу

**Может отсутствовать:** *нет*

**Может повторяться:** *нет*

**Описание.** Данный элемент задаёт путь к каталогу, в котором хранятся описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`.

**Пример.**

```
contests_dir = "/var/ejudge/contests"
```

## 2.9.9 Параметр `show_generation_time`

**Имя параметра:** `show_generation_time`

**Тип содержимого:** *boolean*

**Может отсутствовать:** *да*

**Значение по умолчанию:** *false*

**Может повторяться:** *нет*

**Описание.** Если данный параметр установлен в *true*, в конце генерируемой html-страницы добавляется информация о процессорном времени, затраченном на её генерацию.

**Пример.**

```
show_generation_time
```

## 2.10 Конфигурационный файл турниров `contest.xml`

В данном разделе даётся описание формата файла описания параметров турнира. В системе **ejudge** описание каждого турнира хранится в отдельном XML-файле. Все файлы располагаются в одном каталоге, который задаётся в элементе `contests_dir` для конфигурационных файлов в формате XML или в параметре `contests_dir` для конфигурационных файлов в формате INI. Описание каждого турнира хранится в этом каталоге в файле с именем `N.xml`, где  $N$  — идентификатор (номер) турнира в десятичной записи из 6 знаков с ведущими нулями. Например, информация о турнире с номером 1 хранится в файле `000001.xml`, а информация о турнире с номером 192 — в файле `000192.xml`. Идентификатор турнира назначается администратором системы **ejudge** и должен быть целым числом в диапазоне  $1 \leq N \leq 999999$ . Никакие два турнира в одной инсталляции системы **ejudge** не могут иметь один и тот же идентификатор. Для большей эффективности работы системы **ejudge** рекомендуется нумеровать турниры подряд, начиная с 1. Идентификатор турнира, определяемый по имени файла, в котором хранится его описание, должен совпадать со значением атрибута `id` элемента верхнего уровня `contest` самого файла.

Файл описания турнира должен представлять собой правильно сформированный XML-файл. Элементом верхнего уровня файла является элемент `contest`, который должен присутствовать и не может повторяться. Общая структура вложенности элементов файла приведена в таблице 2.3.

В конфигурационном файле турнира, в частности, задаются дополнительные ограничения на допустимые IP-адреса для всех CGI-программ (элементы ...). Кроме этого, в конфигурационном файле определяются обязательные и необязательные поля анкеты участника.

Анкета участника турнира заполняется пользователем, когда он регистрируется на турнир с помощью CGI-программы [register](#). Участником турнира может быть как команда (для командных соревнований), так и единственный человек. Поэтому анкета участника турнира состоит из нескольких частей. Первую часть составляют данные общего характера, такие как название участника турнира (то есть название команды или имя, под которым в турнире выступает человек), учебное заведение, факультет, город, страна. Вторую часть анкеты составляют персональные данные всех лиц, связанных с командой (в случае командного турнира) или участником. Для командных турниров могут задаваться личные данные членов команды, запасных игроков, тренеров и руководителей команды, а также гостей, приглашаемых командой. Для личных турниров могут задаваться личные данные участника турнира, его тренера и руководителя.

Поскольку количество требуемой информации об участнике может варьироваться от турнира к турниру, так же как и ограничения на число лиц, связанных с участником или командой, в конфигурационном файле турнира задаются поля анкеты, предъявляемые для заполнения. Поля анкеты первой части описываются в элементах `field`, вложенных непосредственно в элемент `contest`. Атрибут `id` элемента в этом случае может принимать значения, перечисленные в таблице 2.4.

Поля второй части анкеты описываются отдельно по каждой категории лиц, связанных с участником. Выделяются пять категорий лиц:

1. Непосредственно игроки (`contestants`). В случае командного турнира это — члены команды, в случае личного турнира это сам участник турнира.
2. Запасные игроки (`reserves`). В случае личного турнира эта категория не имеет смысла.

```

contest
  name
  name_en
  register_access
    ip
  users_access
    ip
  master_access
    ip
  judge_access
    ip
  team_access
    ip
  field
  contestants
    field
  reserves
    field
  coaches
    field
  advisors
    field
  guests
    field
  users_header_file
  users_footer_file
  register_header_file
  register_footer_file
  team_header_file
  team_footer_file
  register_email
  register_url
  team_url
  registration_deadline
  users_head_style
  users_par_style
  users_table_style
  users_verb_style
  register_head_style
  register_par_style
  register_table_style
  team_head_style
  team_par_style

  caps
    cap
  root_dir
  standings_url
  problems_url
  client_flags
  serve_user
  serve_group

```

Таблица 2.3: Структура вложенности элементов конфигурационного файла `contest.xml`

homepage	Домашняя страница пользователя.
inst	Полное название учебного заведения, к которому относится участник.
inst_en	Полное название учебного заведения, к которому относится участник, на английском языке.
instshort	Краткое название учебного заведения.
instshort_en	Краткое название учебного заведения на английском языке.
fac	Полное название факультета, к которому относится участник.
fac_en	Полное название факультета, к которому относится участник, на английском языке.
facshort	Краткое название факультета.
facshort_en	Краткое название факультета на английском языке.
city	Город.
city_en	Название города на английском языке.
country	Страна.
country_en	Название страны на английском языке.

Таблица 2.4: Описание полей анкеты участника, допустимых в первой части анкеты

3. Тренеры (coaches).

4. Руководители (advisors).

5. Прочие (guests).

Для каждой из категории атрибутами соответствующего элемента файла конфигурации турнира задаётся минимальное и максимальное число лиц этой категории. Поля анкеты задаются с помощью вложенных элементов `field`, атрибут `id` которых может принимать значения, перечисленные в таблице 2.5.

Далее рассматриваются все элементы конфигурационного файла `contest.xml`.

firstname	Имя.
firstname_en	Имя в английском написании.
middlename	Отчество.
middlename_en	Отчество в английском написании.
surname	Фамилия.
surname_en	Фамилия в английском написании.
status	Статус: школьник, студент, аспирант, учитель и т. д.
grade	Класс школы или номер курса ВУЗа.
group	Название класса или академической группы.
group_en	Название класса или академической группы в английском написании.
email	Личный адрес e-mail.
homepage	Личная страничка в Интернет.
inst	Полное название учебного заведения (если отличается от заданного для участника в целом).
inst_en	Полное название учебного заведения (если отличается от заданного для участника в целом) на английском языке.
instshort	Краткое название учебного заведения.
instshort_en	Краткое название учебного заведения на английском языке.
fac	Полное название факультета.
fac_en	Полное название факультета на английском языке.
facshort	Краткое название факультета.
facshort_en	Краткое название факультета на английском языке.
occupation	Занимаемая должность.
occupation_en	Занимаемая должность на английском языке.

Таблица 2.5: Описание полей анкеты участника, допустимых во второй части анкеты



## 2.10.1 Элемент `contest`

<b>Имя элемента:</b>	<code>contest</code>
<b>Содержится в:</b>	<i>нет</i>
<b>Может содержать:</b>	<code>name, name_en, register_access, users_access, master_access, judge_access, team_access, field, contestants, reserves, coaches, advisors, guests, users_header_file, users_footer_file, register_header_file, register_footer_file, team_header_file, team_footer_file, register_email, register_url, team_url, registration_deadline, caps, root_dir, standings_url, problems_url, client_flags, serve_user, serve_group, users_head_style, users_par_style, users_table_style, users_verb_style, register_head_style, register_par_style, register_table_style, team_head_style, team_par_style</code>
<b>Атрибуты:</b>	<code>id, autoregister, managed, run_managed, clean_users, disable_team_password</code>
<b>Тип содержимого:</b>	игнорируется
<b>Может отсутствовать:</b>	<i>нет</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Элемент верхнего уровня конфигурационного файла. Элемент должен иметь обязательный атрибут `id`, определяющий идентификатор этого турнира.

### Атрибут `id`

<b>Имя атрибута:</b>	<code>id</code>
<b>Содержится в:</b>	<code>contest</code>
<b>Тип значения:</b>	<i>integer</i>
<b>Может отсутствовать:</b>	<i>нет</i>

**Описание.** Данный атрибут должен содержать идентификатор турнира. Идентификатор турнира представляет собой число в десятичной записи в диапазоне от 1 до 999999 включительно. Идентификатор турнира, определяемого в файле, должен соответствовать имени этого файла.

## Атрибут `autoregister`

Имя атрибута:	<code>autoregister</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>

**Описание.** Данный атрибут устанавливает значение флага авторегистрации. Если атрибут установлен в значение *true*, регистрация каждого нового участника турнира автоматически подтверждается как только он полностью заполнил регистрационную форму. Сразу после этого участник может заходить на личную страницу сдачи решений. Если атрибут установлен в значение *false* (по умолчанию), после того, как участник полностью заполнил регистрационную форму, его регистрации присваивается статус «Ожидает подтверждения». В этом статусе участник не может заходить на свою личную страницу. Подтверждение регистрации может быть сделано привилегированным пользователем турнира, который имеет полномочие `EDIT_REGISTRATION`.

Значение этого атрибута рекомендуется устанавливать в *true* для постоянно открытых турниров, таких как интернет-туры или виртуальные турниры. Значение атрибута *false* позволяет регулировать регистрацию на турнир и может применяться для традиционной формы проведения турниров.

## Атрибут `managed`

Имя атрибута:	<code>managed</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>

**Описание.** Если значение этого атрибута установлено в *true*, программа управления турниром `serve` ставится под управление программы `super-serve`, которая следит одновременно за несколькими турнирами и запускает программу `serve`, как только поступает запрос соответствующему турниру. Если значение атрибута установлено в *false*, данный турнир игнорируется программой `super-serve`.

## Атрибут `run_managed`

Имя атрибута:	<code>run_managed</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>

**Описание.** Если значение этого атрибута установлено в *true*, программа проверки решений `run` ставится под управление программы `super-serve`, которая следит одновременно за несколькими турнирами и запускает программу `run`, как только поступает запрос на проверку решения в соответствующем турнире. Если значение атрибута установлено в *false*, данный турнир игнорируется программой `super-serve`.

## Атрибут `clean_users`

Имя атрибута:	<code>clean_users</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>true</code>

**Описание.** Если значение данного атрибута установлено в `true` (значение по умолчанию), программа `clean-users` будет сканировать базу сдач решений и вопросов судьям текущего турнира. Программа `clean-users` выявляет пользователей, занесённых в базу пользователей, возможно зарегистрировавшихся для участия в каком-либо турнире, но не сделавших ни одной попытки сдачи решения и не задавших ни один вопрос. Такие пользователи помечаются как кандидаты на удаление из базы пользователей. Если значение данного атрибута установлено в `false`, программа `clean-users` игнорирует данный турнир.

## Атрибут `disable_team_password`

Имя атрибута:	<code>disable_team_password</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>

**Описание.** По умолчанию каждый пользователь имеет два пароля: регистрационный и пароль участника. Регистрационный пароль необходим для доступа к CGI-программе `register`, которая позволяет изменять регистрационную информацию и регистрироваться для участия в турнирах. Пароль участника необходим для доступа к CGI-программе `team`, то есть непосредственно для участия в турнире. Если значение атрибута `disable_team_password` равно `true`, специальный пароль участия в турнире отключается. Для доступа к CGI-программе `team` в этом случае используется регистрационный пароль.

Пароли участников обычно генерируются администратором турнира (точнее, пользователем с полномочием `GENERATE_TEAM_PASSWORDS`) перед началом турнира. Для турниров с открытой регистрацией (см. атрибут `autoregister` элемента `contest`) это становится затруднительным. Тогда использование атрибута `disable_team_password` может быть удобным.

## Атрибут `closed`

Имя атрибута:	<code>closed</code>
Содержится в:	<code>contest</code>
Тип значения:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>

**Описание.** Если этот атрибут установлен в `true`, данный турнир считается «закрытым» (оконченным), то есть регистрация на него с помощью CGI-программы `register` невозможна, и запуск CGI-программы `team` для данного турнира также невозможен (будет выведено

сообщение о том, что данный турнир закрыт). Данный атрибут турнира может устанавливаться, когда желательно предотвратить случайные бессмысленные попытки регистрации на турнир и участия в нём.

### 2.10.2 Элемент **name**

Имя элемента:	<b>name</b>
Содержится в:	<a href="#">contest</a>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>
Описание.	Данный элемент позволяет задать имя турнира.
Пример.	

`<name>Турнир пионеров</name>`

### 2.10.3 Элемент **name\_en**

Имя элемента:	<b>name</b>
Содержится в:	<a href="#">contest</a>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать имя турнира на английском языке. Английское имя турнира используется программами **users**, **register** и **team** когда идентификатор языка (`locale_id`) равен 0. Если элемент `name_en` не задан, используется значение элемента `name`, то есть имя турнира на локальном языке (в настоящее время это может быть только русский язык).

**Замечание.** Английское название турнира должно использоваться во всех случаях, когда идентификатор языка в CGI-программах не совпадает с идентификатором языка, указанным в конфигурационном файле турнира. Данная возможность в настоящее время не реализована, поскольку система **ejudge** поддерживает пока только два языка: английский и русский.

**Пример.**

`<name_en>The pioneer's tournament</name_en>`

### 2.10.4 Элемент **register\_access**

Имя элемента:	<b>register_access</b>
Содержится в:	<a href="#">contest</a>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	<a href="#">список IP-ограничений</a>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать ограничения на IP-адреса, с которых возможна регистрация на данный турнир с помощью CGI-программы [register](#). Для подробного описания средств ограничения доступа по IP-адресам системы **ejudge** см. раздел [2.2](#).

## 2.10.5 Элемент `users_access`

Имя элемента:	<code>users_access</code>
Содержится в:	<code>contest</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	<i>список IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать ограничения на IP-адреса, с которых возможен просмотр списка участников и информации об участниках с помощью CGI-программы `users`. Для подробного описания средств ограничения доступа по IP-адресам системы `ejudge` см. раздел [2.2](#).

## 2.10.6 Элемент `master_access`

Имя элемента:	<code>master_access</code>
Содержится в:	<code>contest</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	<i>список IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать ограничения на IP-адреса, с которых возможно администрирование данного турнира с помощью CGI-программы `master`. Для подробного описания средств ограничения доступа по IP-адресам системы `ejudge` см. раздел [2.2](#).

## 2.10.7 Элемент `judge_access`

Имя элемента:	<code>judge_access</code>
Содержится в:	<code>contest</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	<i>список IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать ограничения на IP-адреса, с которых возможно судейство данного турнира с помощью CGI-программы `judge`. Для подробного описания средств ограничения доступа по IP-адресам системы `ejudge` см. раздел [2.2](#).

## 2.10.8 Элемент `team_access`

Имя элемента:	<code>team_access</code>
Содержится в:	<code>contest</code>
Может содержать:	<code>ip</code>
Атрибуты:	<code>default</code>
Тип содержимого:	<i>список IP-ограничений</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать ограничения на IP-адреса, с которых возможно участие в данном турнире с помощью CGI-программы `team`. Для подробного описания средств ограничения доступа по IP-адресам системы `ejudge` см. раздел 2.2.

## 2.10.9 Элемент `ip`

Имя элемента:	<code>ip</code>
Содержится в:	<code>register_access</code> , <code>users_access</code> , <code>master_access</code> , <code>judge_access</code> , <code>team_access</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>allow</code>
Тип содержимого:	<i>IP-ограничение</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>

**Описание.** Данный элемент задаёт одно ограничения на IP-адреса. Для подробного описания средств ограничения доступа по IP-адресам системы `ejudge` см. раздел 2.2.

## 2.10.10 Элемент `field`

Имя элемента:	<code>field</code>
Содержится в:	<code>contest</code> , <code>contestants</code> , <code>reserves</code> , <code>coaches</code> , <code>advisors</code> , <code>guests</code>
Может содержать:	<i>нет</i>
Атрибуты:	<code>id</code> , <code>mandatory</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>да</i>

**Описание.** Элемент `field` позволяет указывать поля ввода, которые будут предъявлены участнику при регистрации на турнир. Если элемент находится непосредственно в элементе `contest`, он позволяет указывать поля ввода общей информации об участнике. В этом случае атрибут `id` должен принимать одно из значений, перечисленных в таблице 2.4. Если элемент находится в элементах `contestants`, `reserves`, `coaches`, `advisors`, и `guests`, с его помощью указываются поля анкеты, заполняемые для соответствующей категории лиц. В этом случае атрибут `id` должен принимать одно из значений, перечисленных в таблице 2.5. С помощью атрибута `mandatory` элемента `field` можно указать, что данное поле ввода является обязательным для заполнения в анкете.

## Атрибут `id`

**Имя атрибута:** `id`  
**Содержится в:** `field`  
**Тип значения:** См. таблицы 2.4 и 2.5.  
**Может отсутствовать:** *нет*

**Описание.** С помощью данного атрибута указывается поле ввода анкеты. Если элемент `field` непосредственно вложен в элемент `contest`, данный атрибут должен принимать одно из значений, перечисленных в таблице 2.4. Если элемент `field` вложен в элементы `contestants`, `reserves`, `coaches`, `advisors`, и `guests`, данный атрибут должен принимать одно из значений, перечисленных в таблице 2.5.

## Атрибут `mandatory`

**Имя атрибута:** `mandatory`  
**Содержится в:** `field`  
**Тип значения:** `boolean`  
**Может отсутствовать:** *да*  
**Значение по умолчанию:** `false`

**Описание.** Данный атрибут позволяет указать, что поле ввода анкеты, задаваемое с помощью данного элемента `field`, обязательно к заполнению.



## 2.10.11 Элементы `contestants`, `reserves`, `coaches`, `advisors`, `guests`

Имя элемента:	<code>contestants</code> , <code>reserves</code> , <code>coaches</code> , <code>advisors</code> , <code>guests</code>
Содержится в:	<code>contest</code>
Может содержать:	<code>field</code>
Атрибуты:	<code>min</code> , <code>max</code> , <code>initial</code>
Тип содержимого:	игнорируется
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Описание нескольких элементов объединено в один раздел, так как эти элементы очень близки друг другу. Данные элементы позволяют задавать ограничения на количество лиц соответствующей категории. Кроме того, данные элементы содержат в себе элементы `field`, задающие поля ввода регистрационной анкеты. Элемент `contestants` соответствует категории игроков турнира, элемент `reserves` соответствует категории запасных игроков, элемент `coaches` — категории тренеров, элемент `advisors` — категории руководителей, и элемент `guests` — прочим.

Атрибут `min` позволяет задать минимальное число лиц в данной категории. Например, минимальное количество игроков одной команды (`contestants`) в командном турнире может быть установлено в 2 или 3, а в личном турнире — в 1. Если у каждого участника должен быть тренер, минимальное число лиц этой категории (`coaches`) должно быть установлено в 1. Атрибут `max` позволяет задать максимальное число лиц в данной категории. Например, максимальное количество игроков одной команды в командном турнире равно 3, а в личном турнире — 1. Атрибут `initial` позволяет задать, формы ввода для скольких лиц в данной категории будут выведены при регистрации на турнир, когда формы печатаются для данного пользователя первый раз.

### Атрибут `min`

Имя атрибута:	<code>min</code>
Содержится в:	<code>contestants</code> , <code>reserves</code> , <code>coaches</code> , <code>advisors</code> , <code>guests</code>
Тип значения:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	0
Описание:	Минимальное количество лиц в данной категории.

### Атрибут `max`

Имя атрибута:	<code>max</code>
Содержится в:	<code>contestants</code> , <code>reserves</code> , <code>coaches</code> , <code>advisors</code> , <code>guests</code>
Тип значения:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	0
Описание:	Максимальное количество лиц в данной категории.

## Атрибут **initial**

<b>Имя атрибута:</b>	<b>initial</b>
<b>Содержится в:</b>	contestants, reserves, coaches, advisors, guests
<b>Тип значения:</b>	integer
<b>Может отсутствовать:</b>	да
<b>Значение по умолчанию:</b>	0
<b>Описание:</b>	Начальное количество лиц в данной категории.

### Пример

```
<contestants min="2" max="3">
  <field id="firstname" mandatory="yes"/>
  <field id="middlename"/>
  <field id="surname" mandatory="yes"/>
  <field id="status" mandatory="yes"/>
  <field id="grade" mandatory="yes"/>
  <field id="group"/>
</contestants>
<reserves min="0" max="1">
  <field id="firstname" mandatory="yes"/>
  <field id="middlename"/>
  <field id="surname" mandatory="yes"/>
  <field id="status" mandatory="yes"/>
  <field id="grade" mandatory="yes"/>
  <field id="group"/>
</reserves>
<coaches min="1" max="1">
  <field id="firstname" mandatory="yes"/>
  <field id="middlename"/>
  <field id="surname" mandatory="yes"/>
  <field id="status" mandatory="yes"/>
  <field id="occupation" mandatory="yes"/>
</coaches>
<advisors min="0" max="1">
  <field id="firstname" mandatory="yes"/>
  <field id="middlename"/>
  <field id="surname" mandatory="yes"/>
  <field id="status" mandatory="yes"/>
  <field id="occupation" mandatory="yes"/>
</advisors>
```

## 2.10.12 Элемент `users_header_file`

Имя элемента:	<code>users_header_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «шапку» html-файла, генерируемого CGI-программой `users`. Если элемент задан, то содержимое указанного файла будет выведено в начале генерируемого файла. Если элемент не задан, генерируется заголовок по умолчанию, включающий название турнира и кодировку страницы. В файле-заголовке допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице 2.6.

**Пример.**

```
<users_header_file>/home/judges/20/conf/header.shtml</users_header_file>
```

**Замечание.** Если в файле-заголовке используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.13 Элемент `users_footer_file`

Имя элемента:	<code>users_footer_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «подвал» html-файла, генерируемого CGI-программой `users`. Если элемент задан, то содержимое указанного файла будет выведено в конце генерируемого файла. Если элемент не задан, генерируется подвал страницы по умолчанию, включающий информацию об авторских правах и версии системы `ejudge`. В файле-подвале допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице 2.7.

**Пример.**

```
<users_footer_file>/home/judges/20/conf/footer.shtml</users_footer_file>
```

**Замечание.** Если в файле-подвале используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.14 Элемент `register_header_file`

Имя элемента:	<code>register_header_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «шапку» html-файла, генерируемого CGI-программой [register](#). Если элемент задан, то содержимое указанного файла будет выведено в начале генерируемого файла. Если элемент не задан, генерируется заголовок по умолчанию, включающий название турнира и кодировку страницы. В файле-заголовке допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице [2.6](#).

**Пример.**

```
<register_header_file>/home/judges/20/conf/header.shtml</register_header_file>
```

**Замечание.** Если в файле-заголовке используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.15 Элемент `register_footer_file`

Имя элемента:	<code>register_footer_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «подвал» html-файла, генерируемого CGI-программой [register](#). Если элемент задан, то содержимое указанного файла будет выведено в конце генерируемого файла. Если элемент не задан, генерируется подвал страницы по умолчанию, включающий информацию об авторских правах и версии системы `ejudge`. В файле-подвале допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице [2.7](#).

**Пример.**

```
<register_footer_file>/home/judges/20/conf/footer.shtml</register_footer_file>
```

**Замечание.** Если в файле-подвале используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.16 Элемент `team_header_file`

Имя элемента:	<code>team_header_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «шапку» html-файла, генерируемого CGI-программой `team`. Если элемент задан, то содержимое указанного файла будет выведено в начале генерируемого файла. Если элемент не задан, генерируется заголовок по умолчанию, включающий название турнира и кодировку страницы. В файле-заголовке допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице 2.6.

**Пример.**

```
<team_header_file>/home/judges/20/conf/header.shtml</team_header_file>
```

**Замечание.** Если в файле-заголовке используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.17 Элемент `team_footer_file`

Имя элемента:	<code>team_footer_file</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	путь к файлу
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать путь к файлу, который содержит «подвал» html-файла, генерируемого CGI-программой `team`. Если элемент задан, то содержимое указанного файла будет выведено в конце генерируемого файла. Если элемент не задан, генерируется подвал страницы по умолчанию, включающий информацию об авторских правах и версии системы `ejudge`. В файле-подвале допускаются специальные конструкции, начинающиеся со знака % («процент»), перечисленные в таблице 2.7.

**Пример.**

```
<team_footer_file>/home/judges/20/conf/footer.shtml</team_footer_file>
```

**Замечание.** Если в файле-подвале используются server-side includes (SSI), поддержка SSI в CGI-программах должна быть включена в используемом http-сервере. Для дальнейшей информации обратитесь к документации на Ваш http-сервер.

## 2.10.18 Элемент `users_head_style`

`users_head_style`

**Имя элемента:** `users_head_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `h2`

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет установить тег языка HTML, который используется для выделения названий секций в html-странице, генерируемой CGI-программой `users`. Если значение этого элемента не установлено, используется тег `h2`, то есть заголовки в генерируемом html-документе будут иметь примерно следующий вид.

```
<h2>Список участников турнира X</h2>
```

### Пример.

```
<users_head_style>h3</users_head_style>
```

Такое значение данной конфигурационной переменной приведёт к тому, что тот же самый заголовок будет иметь следующий вид.

```
<h3>Список участников турнира X</h3>
```

**Замечание.** В текущей реализации системы `ejudge` отсутствует возможность задать отдельно открывающий и закрывающий тег. И для того, и для другого будет использоваться значение данной переменной. Поэтому указание ещё каких-либо дополнительных атрибутов тега заголовка приведёт к тому, что дополнительные атрибуты продублируются в закрывающем теге, приводя, строго говоря, к неправильному html.

## 2.10.19 Элемент `users_par_style`

`users_par_style`

**Имя элемента:** `users_par_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тега `<p>` в html-страницах, генерируемых программой `users`. Значение элемента `users_par_style` добавляется в открывающий тег `<p>`. Закрывающий тег `</p>` остаётся без изменений.

**Пример.**

```
<users_par_style> class="common"</users_par_style>
```

Такое значение конфигурационной переменной `users_par_style` приведёт к тому, что в генерируемых CGI-программой `users` html-страницах все абзацы будут помечены следующим образом:

```
<p class="common">A paragraph</p>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `users_par_style`. Если бы он отсутствовал, тег `p` и атрибут `class` слились в одно слово `pclass`.

## 2.10.20 Элемент `users_table_style`

`users_table_style`

**Имя элемента:** `users_table_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тегов `<th>` и `<td>` в html-странице, содержащей таблицу всех участников, зарегистрированных на турнир, генерируемой программой `users`. Значение элемента `users_table_style` добавляется в открывающие теги `<th>` и `<td>`. Закрывающие теги `</th>` и `</td>` остаются без изменений.

**Пример.**

```
<users_table_style> class="reg"</users_table_style>
```

Такое значение конфигурационной переменной `users_table_style` приведёт к тому, что в генерируемой CGI-программой `users` таблице участников турнира все заголовки столбцов и ячейки таблицы будут помечены следующим образом:

```
<td class="reg">User</td>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `users_table_style`. Если бы он отсутствовал, тег `td` и атрибут `class` слились в одно слово `tdclass`.



## 2.10.21 Элемент `users_verb_style`

`users_verb_style`

**Имя элемента:** `users_verb_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тегов `<th>` и `<td>` в html-странице, содержащей подробную информацию об одном участнике и генерируемой программой `users`. Значение элемента `users_verb_style` добавляется в открывающие теги `<th>` и `<td>`. Закрывающие теги `</th>` и `</td>` остаются без изменений.

**Пример.**

```
<users_verb_style> class="verb"</users_verb_style>
```

Такое значение конфигурационной переменной `users_verb_style` приведёт к тому, что в генерируемой CGI-программой `users` html-странице с подробной информацией об участнике все заголовки столбцов и ячейки таблицы будут помечены следующим образом:

```
<td class="verb">User</td>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `users_verb_style`. Если бы он отсутствовал, тег `td` и атрибут `class` слились в одно слово `tdclass`.

## 2.10.22 Элемент `register_head_style`

`register_head_style`

**Имя элемента:** `register_head_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `h2`

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет установить тег языка HTML, который используется для выделения названий секций в html-странице, генерируемой CGI-программой `register`. Если значение этого элемента не установлено, используется тег `h2`, то есть заголовки в генерируемом html-документе будут иметь примерно следующий вид.

```
<h2>Персональная информация пользователя X</h2>
```

### Пример.

```
<register_head_style>h3</register_head_style>
```

Такое значение данной конфигурационной переменной приведёт к тому, что тот же самый заголовок будет иметь следующий вид.

```
<h3>Персональная информация пользователя X</h3>
```

**Замечание.** В текущей версии системы `ejudge` отсутствует возможность задать отдельно открывающий и закрывающий тег. И для того, и для другого будет использоваться значение данной переменной. Поэтому указание ещё каких-либо дополнительных атрибутов тега заголовка приведёт к тому, что дополнительные атрибуты продублируются в закрывающем теге, приводя, строго говоря, к неправильному html.

## 2.10.23 Элемент `register_par_style`

`register_par_style`

**Имя элемента:** `register_par_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тега `<p>` в html-страницах, генерируемых программой `register`. Значение элемента `register_par_style` добавляется в открывающий тег `<p>`. Закрывающий тег `</p>` остаётся без изменений.

**Пример.**

```
<register_par_style> class="common"</register_par_style>
```

Такое значение конфигурационной переменной `register_par_style` приведёт к тому, что в генерируемых CGI-программой `register` html-страницах все абзацы будут помечены следующим образом:

```
<p class="common">A paragraph</p>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `register_par_style`. Если бы он отсутствовал, тег `p` и атрибут `class` слились в одно слово `pclass`.

## 2.10.24 Элемент `register_table_style`

`register_table_style`

**Имя элемента:** `register_table_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тегов `<th>` и `<td>` в html-странице, содержащей таблицу всех турниров, на которые зарегистрировался данный пользователь, генерируемой программой `register`. Значение элемента `register_table_style` добавляется в открывающие теги `<th>` и `<td>`. Закрывающие теги `</th>` и `</td>` остаются без изменений.

**Пример.**

```
<register_table_style> class="reg"</register_table_style>
```

Такое значение конфигурационной переменной `register_table_style` приведёт к тому, что в генерируемой CGI-программой `register` таблице участников турнира все заголовки столбцов и ячейки таблицы будут помечены следующим образом:

```
<td class="reg">User</td>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `register_table_style`. Если бы он отсутствовал, тег `td` и атрибут `class` слились в одно слово `tdclass`.

## 2.10.25 Элемент `team_head_style`

`team_head_style`

**Имя элемента:** `team_head_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `h2`

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет установить тег языка HTML, который используется для выделения названий секций в html-странице, генерируемой CGI-программой `team`. Если значение этого элемента не установлено, используется тег `h2`, то есть заголовки в генерируемом html-документе будут иметь примерно следующий вид.

```
<h2>Персональная страница участника X</h2>
```

### **Пример.**

```
<team_head_style>h3</team_head_style>
```

Такое значение данной конфигурационной переменной приведёт к тому, что тот же самый заголовок будет иметь следующий вид.

```
<h3>Персональная страница участника X</h3>
```

**Замечание.** В текущей версии системы `ejudge` отсутствует возможность задать отдельно открывающий и закрывающий тег. И для того, и для другого будет использоваться значение данной переменной. Поэтому указание ещё каких-либо дополнительных атрибутов тега заголовка приведёт к тому, что дополнительные атрибуты продублируются в закрывающем теге, приводя, строго говоря, к неправильному html.

## 2.10.26 Элемент `team_par_style`

`team_par_style`

**Имя элемента:** `team_par_style`

**Содержится в:** `contest`

**Может содержать:** *нет*

**Атрибуты:** *нет*

**Тип содержимого:** строка

**Может отсутствовать:** *да*

**Значение по умолчанию:** `""` (пустая строка)

**Может повторяться:** *нет*

**Описание.** Данный элемент позволяет задать параметры тега `<p>` в html-страницах, генерируемых программой `team`. Значение элемента `team_par_style` добавляется в открывающий тег `<p>`. Закрывающий тег `</p>` остаётся без изменений.

**Пример.**

```
<team_par_style> class="common"</team_par_style>
```

Такое значение конфигурационной переменной `team_par_style` приведёт к тому, что в генерируемых CGI-программой `team` html-страницах все абзацы будут помечены следующим образом:

```
<p class="common">A paragraph</p>
```

Обратите внимание на пробел в начале строки значения конфигурационной переменной `team_par_style`. Если бы он отсутствовал, тег `p` и атрибут `class` слились в одно слово `pclass`.

## 2.10.27 Элемент `register_email`

Имя элемента:	<code>register_email</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	e-mail адрес
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать адрес отправителя (поле From) электронного письма, посылаемого пользователю при регистрации в системе. Это письмо посылается по адресу, который был задан пользователем в регистрационной анкете и содержит регистрационное имя, пароль пользователя, а также URL, для дальнейшей регистрации в системе. Если данный элемент в конфигурационном файле турнира не задан, используется значение элемента `register_email` конфигурационного файла `ejudge.xml`. Если и тот элемент не задан, используется адрес по умолчанию `team@contest.cmc.msu.ru`.

**Пример.**

```
<register_email>register@supercontest.ru</register_email>
```

## 2.10.28 Элемент `register_url`

Имя элемента:	<code>register_url</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	URL
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать URL CGI-программы `register` для дальнейшей регистрации в системе `ejudge`. Этот URL будет помещён в текст письма-уведомления, посылаемого пользователю при регистрации в системе. Если в конфигурационном файле турнира `contest.xml` данный элемент не задан, используется URL, заданный в элементе `register_url` конфигурационного файла `ejudge.xml`. Если и тот элемент не задан, используется URL по умолчанию `http://contest.cmc.msu.ru/cgi-bin/register`. К URL, заданному в любом из конфигурационных файлов или по умолчанию, автоматически добавляется параметр `contest_id` со значением, равным идентификатору данного турнира, параметр `locale_id` со значением, равным идентификатору выбранного языка интерфейса, параметр `login` со значением, равным регистрационному имени нового пользователя, и параметр `action` со значением 3 для перехода непосредственно к странице входа в систему. Таким образом, полностью сформированный URL для примера ниже может выглядеть следующим образом:

```
http://www.supercontest.ru/cgi-bin/register?action=3&login=user&contest_id=4&locale_id=1
```

**Пример.**

```
<register_url>http://www.supercontest.ru/cgi-bin/register</register_url>
```

## 2.10.29 Элемент `team_url`

Имя элемента:	<code>team_url</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	URL
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задать URL для CGI-программы `team`, которая позволяет пользователю непосредственно участвовать в турнире. Если данный элемент задан, то ссылка на указанный URL будет выведена на личной странице участника после успешной регистрации участника в данном турнире, то есть когда статус регистрации участника установлен в ОК. Если элемент не задан, ссылка не демонстрируется.

**Пример.**

```
<team_url>http://www.supercontest.ru/cgi-bin/team</team_url>
```

## 2.10.30 Элемент `registration_deadline`

Имя элемента:	<code>registration_deadline</code>
Содержится в:	<code>contest</code>
Может содержать:	<i>нет</i>
Атрибуты:	<i>нет</i>
Тип содержимого:	<i>астрономическое время</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данный элемент позволяет задавать крайний срок приёма регистрационных заявок на данный турнир. Когда астрономическое время станет больше указанного, заявки на участие в данном турнире приниматься не будут, и информация о возможности регистрации на данный турнир не будет выводиться на личной странице участников. Если данный элемент не задан, регистрация не имеет ограничений во времени.

Дата должна задаваться в формате *Year/Month/Day Hour:Min:Sec*. Здесь *Year* — четырёхзначный номер года от 1906 до 2038 (диапазон представимых дат во внутреннем формате для систем POSIX). *Month* — номер месяца от 1 до 12, *Day* — номер дня в месяце от 1 до 31, *Hour* — час суток от 0 до 23, *Min* — минуты от 0 до 59, *Sec* — секунды от 0 до 59. Элементы даты могут опускаться, начиная с последнего. В этом случае подразумевается минимальное значение. То есть, в спецификации календарного времени могут быть опущены секунды (например, 2003/02/28 14:12), секунды и минуты (например, 2003/02/28 14), секунды, минуты и часы (например, 2003/02/28). Поскольку для конвертации астрономического времени во внутреннее представление используется функция `mktime(3)`, проверка на допустимость даты, например, 2003/02/30, не производится. Такая дата окажется автоматически преобразованной в 2003/03/02.

**Пример.**

```
<registration_deadline>2003/10/05 00:00:00</registration_deadline>
```



%L	Заменяется на идентификатор языкового окружения, выбранного пользователем. Английскому языку соответствует идентификатор 0, а русскому — 1.
%C	Заменяется на кодировку (charset) страницы, например, koi8-r.
%T	Заменяется на тип содержимого (content-type) страницы, например, text/html.
%H	Заменяется на заголовок страницы, который может включать в себя название турнира и/или имя пользователя.
%%	Заменяется на один знак %.

Таблица 2.6: Специальные конструкции подстановки в файлах-заголовках

%R	Заменяется на информацию об авторских правах, правах на распространение (GNU GPL) и версии системы <b>ejudge</b> .
%%	Заменяется на один знак %.

Таблица 2.7: Специальные конструкции подстановки в файлах-подвалах

## 2.11 Конфигурационный файл сервера турнира `serve.cfg`

В данной главе описывается формат конфигурационного файла сервера турнира. Он используется программами `serve`, `compile`, `run`.

### 2.11.1 Общая структура

Конфигурационный файл `serve.cfg` представляет собой текстовый файл в формате, похожем на формат `.ini`-файлов. Комментарии в конфигурационном файле начинаются с символа `#` или `;` и продолжаются до конца строки. Использование символа начала комментария `#` не рекомендуется, если конфигурационный файл предполагается препроцессировать перед чтением (см. ключ `-E` программ `serve`, `compile`, `run`). В этом случае начало комментария, например

```
# include some more information
```

может совпасть с какой-либо директивой препроцессора Си<sup>1</sup>. Кроме того, в конфигурационном файле игнорируются пустые строки.

Конфигурационный файл разбивается на секции. Каждая секция начинается со строки вида

```
[SECTION_NAME]
```

где `SECTION_NAME` — имя секции, например `problem`. В начале конфигурационного файла находится секция глобальных параметров, которая не имеет заголовка.

Каждый конфигурационный параметр задаётся в отдельной строке в одной из следующих форм.

```
NAME = VALUE
```

Перед именем конфигурационной переменной, перед и после знака равенства и после значения параметра может находиться произвольное число пробельных символов. Имя переменной `NAME` состоит из латинских заглавных и строчных букв, цифр, знака подчёркивания. Значение `VALUE` состоит из произвольных непробельных символов. Данная форма задания конфигурационных переменных может использоваться, когда значение переменной не содержит пробельных символов.

```
NAME = "VALUE"
```

Перед именем конфигурационной переменной, вокруг знака равенства и после закрывающей кавычки может находиться произвольное количество пробельных символов. Имя переменной `NAME` состоит из латинских заглавных и строчных букв, цифр, знака подчёркивания. Значение конфигурационной переменной `VALUE` может содержать произвольные символы (включая пробельные) кроме символа двойной кавычки. Данная форма задания конфигурационной переменной может использоваться, если значение переменной содержит пробельные символы, но не содержит кавычек.

```
NAME
```

---

<sup>1</sup>Судя по всему, использование препроцессора — не очень удачная возможность. Текущая версия системы `ejudge` позволяет в большинстве случаев не использовать препроцессор.

Перед именем конфигурационной переменной и после него может находиться произвольное количество пробельных символов. Данная форма задания параметра эквивалента форме

```
NAME = 1
```

Конфигурационный файл `serve.cfg` состоит из глобальной секции, в которой задаются значения глобальных конфигурационных параметров, одной или нескольких секций описания задач `problem`, одной или нескольких секций описания поддерживаемых языков программирования `language` и одной или нескольких секций описания процедуры тестирования решений `tester`.

## 2.11.2 Форматные подстановки

Некоторые конфигурационные переменные определяют не непосредственно значения, используемые при работе сервера турнира, а *шаблоны* значений. Например, конфигурационная переменная `team_info_url` определяет шаблон URL, который используется при генерации строки таблицы результатов для каждой команды. Этот URL содержит ссылку на страницу с подробной информацией о соответствующем участнике. Поскольку URL зависит от некоторых атрибутов участника (например, от его идентификатора), шаблон URL перед использованием должен быть подвергнут преобразованию.

Другой случай, когда выполняется подстановка шаблонов, возникает при наследовании свойств задачи или тестера от абстрактной задачи или тестера соответственно. Например, конфигурационная переменная `input_name` описания задачи задаёт имя файла с входными данными для решения задачи. Это имя файла может быть как фиксированным (например, `input.txt`), так и зависеть от имени задачи (например, `a.in`, `b.in` и т. д.). В последнем случае в описании абстрактной задачи удобно указать шаблон имени входного файла, который автоматически будет раскрыт при наследовании свойств абстрактной задачи конкретной задачей.

В обоих случаях в необходимый момент выполняются *форматные подстановки*, которые описываются ниже. Аргументами форматной подстановки являются: форматная строка, определяющая выполняемые подстановки; информация о текущей задаче (если доступна); информация о текущей команде (если доступна); информация о текущем тестере (если доступна). Результат форматной подстановки записывается в выходную строку.

Форматная строка состоит из символов, копируемых в выходную строку без изменений, и спецификаций форматного преобразования. Спецификация форматного преобразования начинается со знака «процент» (%) и имеет следующий вид:

```
% [<флаги>] [<ширина>] [ .<точность>] <спецификатор>
```

Здесь в квадратных скобках записаны необязательные элементы спецификации.

Элемент <флаги> позволяет использовать дополнительные возможности преобразования. Элемент <флаги> — это последовательность односимвольных флагов, перечисленных в таблице 2.8. Если в одной спецификации форматного преобразования указаны конфликтующие флаги (например, `l` и `u`), действует флаг, указанный последним.

Элемент <ширина> задаёт *минимальную* ширину поля. Элемент представляет собой последовательность цифр, формирующих целое число в десятичной записи. Данный элемент работает аналогично элементу, задающему ширину в форматном преобразовании в функциях семейства `printf` языка Си. Если не указаны флаги `r`, `0` или `s`, форматная строка, полученная в результате форматного преобразования, выравнивается по левому краю. Если длина

e	Если форматное преобразование даёт пустую строку, она заменяется на строку <code>&amp;nbsp;</code> . Этот флаг может использоваться для генерации содержимого ячеек html-таблиц.
u	Преобразовать строку, полученную в результате форматного преобразования, к верхнему регистру. Например, если короткое имя ( <code>short_name</code> ) задач в турнире имеет вид <code>a, b</code> и т. д., а URL, по которому находятся их условия, имеет вид <code>A.html</code> и т. д., то флаг <code>u</code> вместе с преобразованием <code>Ps</code> (то есть спецификация форматного преобразования <code>%uPs</code> ) даст необходимый результат.
l	Преобразовать строку, полученную в результате форматного преобразования, к нижнему регистру. Например, если короткое имя задачи ( <code>short_name</code> ) задач в турнире имеет вид <code>A, B</code> и т. д., а имя входного файла — <code>a.in</code> и т. д., то спецификация форматного преобразования <code>%lPs</code> даст необходимый результат.
r	Выровнять строку, полученную в результате форматного преобразования, по правому краю, дополняя её слева пробелами при необходимости.
0	Выровнять строку, полученную в результате форматного преобразования, по правому краю, дополняя её слева необходимым количеством нулей.
c	Выровнять строку, полученную в результате форматного преобразования, по центру, дополняя её слева и справа пробелами при необходимости.

Таблица 2.8: Флаги спецификации форматного преобразования

строки, полученной в результате форматного преобразования и отсечения по максимальной длине, задаваемого элементом <точность>, меньше указанной в элементе <ширина>, строка дополняется справа необходимым количеством пробелов. Если указан какой-либо из флагов r, 0 или c, и длина строки, полученной в результате форматного преобразования и отсечения по максимальной длине, задаваемого элементом <точность>, меньше указанной в элементе <ширина>, строка выравнивается согласно указанному флагу. Если длина строки, полученной в результате форматного преобразования и отсечения по максимальной длине, задаваемого элементом <точность>, больше указанной в элементе <ширина>, никакого выравнивания не происходит. Если элемент <ширина> не указан, его значение подразумевается равным нулю.

Элемент <точность> задаёт максимальную длину строки, получаемой в результате раскрытия спецификатора форматного преобразования. Элемент представляет собой последовательность цифр, формирующих целое число в десятичной записи. Данный элемент работает аналогично элементу, задающему точность в форматном преобразовании в функциях семейства printf языка Си. Раскрытие спецификатора форматного преобразования даёт строку, которая подвергается ограничению на максимальную длину. Например, спецификатор форматного преобразования Ps раскрывается в строку короткого имени (short\_name) текущей задачи. Если результат раскрытия спецификатора длиннее, чем указано в элементе <точность>, он обрезается по заданной длине. Если результат раскрытия спецификатора короче, чем указано в элементе <точность>, он сохраняется неизменным. Строка, полученная после применения ограничения на максимальную длину, подвергается преобразованиям выравнивания до минимальной длины, как описано выше. Если элемент <точность> не задан, его значение подразумевается неограниченным.

Элемент <спецификатор> (спецификатор форматного преобразования) указывает данные, которые подставляются вместо спецификации форматного преобразования. Спецификатор состоит из двух букв, где первая буква — это секция, из которой берётся информация: P — текущая задача, T — текущий тестер, M — текущий участник. Вторая буква определяет поле секции. Все поддерживаемые спецификаторы форматного преобразования перечислены в таблице 2.9.

### 2.11.3 Список глобальных параметров

Название	Стр.	Описание
a2ps_args	158	Дополнительные аргументы программы a2ps.
a2ps_path	157	Путь к программе a2ps.
accept_sound	117	Звуковой файл, проигрываемый при успешной сдаче решения.
archive_dir	97	Каталог для хранения архивов турнира.
auto_short_problem_name	133	Флаг автоматической генерации короткого имени задачи.
autoupdate_standings	120	Флаг автоматического обновления таблицы результатов.
board_fog_time	85	Время заморозки таблицы результатов.
board_unfog_time	86	Время разморозки таблицы результатов.
charset	90	Кодировка генерируемых html-страниц.
checker_dir	96	Каталог для запуска проверяемых программ.
<i>Продолжение на следующей странице</i>		

Название	Стр.	Описание
<code>checker_real_time_limit</code>	134	Ограничение времени на работу проверяющей программы.
<code>clar_archive_dir</code>	98	Каталог для хранения сообщений участников и судей.
<code>clar_log_file</code>	110	Файл журнала сообщений участников и судей.
<code>compile_dir</code>	105	Каталог обмена программ <code>serve</code> и <code>compile</code> .
<code>compile_real_time_limit</code>	134	Ограничение времени на компиляцию программ.
<code>compile_work_dir</code>	102	Рабочий каталог программы <code>compile</code> .
<code>conf_dir</code>	93	Каталог конфигурационных файлов.
<code>contest_id</code>	82	Идентификатор турнира.
<code>contests_dir</code>	82	Каталог описания турниров.
<code>contest_time</code>	84	Продолжительность турнира.
<code>corr_dir</code>	95	Каталог с правильными ответами к тестам.
<code>corr_sfx</code>	116	Суффикс файлов с правильными ответами к тестам.
<code>cr_serialization_key</code>	110	Ключ сериализационного семафора.
<code>diff_path</code>	159	Путь к программе сравнения файлов <code>diff</code> .
<code>disable_auto_testing</code>	138	Флаг отключения автоматического тестирования.
<code>disable_clars</code>	130	Флаг полного запрета сообщений.
<code>disable_team_clars</code>	131	Флаг запрета вопросов от участников турнира.
<code>disable_testing</code>	156	Флаг отключения тестирования.
<code>enable_continue</code>	128	Флаг разрешения продолжения турнира.
<code>enable_l10n</code>	91	Флаг включения локализации сообщений.
<code>enable_printing</code>	156	Флаг разрешения печати на принтер участниками турнира.
<code>enable_report_upload</code>	159	Флаг разрешения загрузки протокола тестирования привилегированным пользователем.
<code>enable_runlog_merge</code>	138	Флаг включения средств слияния журнала посылок.
<code>ignore_compile_errors</code>	131	Флаг игнорирования ошибок компиляции.
<code>ignore_duplicated_runs</code>	132	Флаг игнорирования дублирующих посылок.
<code>inactivity_timeout</code>	111	Тайм-аут программы <code>serve</code> .
<code>info_dir</code>	140	Каталог с дополнительной информацией о тестах.
<code>info_sfx</code>	142	Суффикс файлов с дополнительной информацией о тестах.
<code>internal_sound</code>	119	Звук, проигрываемый при внутренней ошибке тестирования.
<code>l10n_dir</code>	91	Каталог файлов локализации.
<code>lpr_args</code>	159	Дополнительные аргументы программы <code>lpr</code> .
<code>lpr_path</code>	158	Путь к программе <code>lpr</code> .
<code>max_clar_num</code>	89	Квота вопросов от одной команды.
<code>max_clar_size</code>	88	Максимальный размер одного вопроса от команды.
<code>max_clar_total</code>	89	Квота суммарного размера вопросов от одной команды.
<code>max_cmd_length</code>	139	Максимальный размер включаемых в протокол тестирования аргументов командной строки.

*Продолжение на следующей странице*

Название	Стр.	Описание
<code>max_file_length</code>	133	Максимальная длина включаемого в протокол тестирования файла.
<code>max_line_length</code>	132	Максимальная длина строки в протоколе тестирования
<code>max_run_num</code>	88	Квота количества решений от одной команды.
<code>max_run_size</code>	87	Максимальный размер одного решения от команды.
<code>max_run_total</code>	87	Квота суммарного размера решений от одной команды.
<code>min_gzip_size</code>	153	Минимальный размер архивного файла для сжатия.
<code>plog_file_name</code>	123	Имя файла с публичным журналом посылок.
<code>plog_footer_file</code>	125	Файл-подвал публичного журнала посылок.
<code>plog_header_file</code>	124	Файл-заголовок публичного журнала посылок.
<code>plog_update_time</code>	124	Период обновления публичного журнала посылок.
<code>presentation_sound</code>	119	Звук, проигрываемый при presentation error.
<code>priority_adjustment</code>	160	Базовый приоритет тестирования.
<code>prob_info_url</code>	126	Шаблон URL текста задачи.
<code>prune_empty_users</code>	143	Флаг игнорирования «пустых» пользователей.
<code>report_archive_dir</code>	99	Каталог архива протоколов тестирования.
<code>report_error_code</code>	128	Флаг разрешения печати кода возврата в протокол участника.
<code>root_dir</code>	92	Корневой каталог файлов турнира.
<code>run_archive_dir</code>	98	Каталог архива посылок.
<code>run_check_dir</code>	104	Рабочий каталог для проверки решений.
<code>run_dir</code>	107	Каталог обмена программ <code>serve</code> и <code>run</code> .
<code>run_log_file</code>	109	Файл журнала посылок.
<code>runtime_sound</code>	117	Звук, проигрываемый при runtime error.
<code>run_work_dir</code>	103	Рабочий каталог программы <code>run</code> .
<code>score_system</code>	112	Тип турнира.
<code>script_dir</code>	93	Каталог скриптов компиляции и запуска.
<code>serve_sleep_time</code>	84	Период опроса каталогов обмена программой <code>serve</code> .
<code>serve_socket</code>	109	UNIX-сокет команд программе <code>serve</code> .
<code>show_astr_time</code>	129	Флаг астрономического времени при печати журнала посылок.
<code>show_deadline</code>	135	Флаг печати крайнего времени сдачи в меню задач.
<code>sleep_time</code>	83	Период опроса каталогов обмена.
<code>socket_path</code>	83	UNIX-сокет для связи с программой <code>userlist-server</code> .
<code>sound_player</code>	116	Программа, которая проигрывает звуковые файлы.
<code>stand2_file_name</code>	122	Имя вторичного файла с таблицей текущих результатов.
<code>stand2_footer_file</code>	123	Файл-подвал вторичного файла таблицы текущих результатов.
<code>stand2_header_file</code>	122	Файл-заголовок вторичного файла таблицы текущих результатов.
<code>standings_charset</code>	90	Кодировка файла таблицы результатов.
<code>standings_file_name</code>	120	Файл основной таблицы результатов.

*Продолжение на следующей странице*

Название	Стр.	Описание
<code>standings_locale</code>	92	Язык файла таблицы результатов.
<code>stand_extra_attr</code>	148	Атрибуты дополнительной пользовательской колонки таблицы результатов.
<code>stand_extra_format</code>	148	Формат вывода в дополнительной пользовательской колонке таблицы результатов.
<code>stand_extra_legend</code>	148	Заголовок дополнительной пользовательской колонки таблицы результатов.
<code>stand_footer_file</code>	121	Файл-подвал основного файла таблицы текущих результатов.
<code>stand_freeze_time</code>	85	Синоним для <code>board_fog_time</code> .
<code>stand_header_file</code>	121	Файл-заголовок основного файла таблицы текущих результатов.
<code>stand_melt_time</code>	86	Синоним для <code>board_unfog_time</code> .
<code>stand_penalty_attr</code>	147	Атрибуты колонки «штраф» таблицы текущих результатов (в режиме турнира <i>ACM</i> ).
<code>stand_place_attr</code>	144	Атрибуты колонки «место» таблицы текущих результатов.
<code>stand_prob_attr</code>	145	Атрибуты колонок «задача» таблицы текущих результатов.
<code>stand_r_row_attr</code>	150	Атрибуты строки таблицы текущих результатов для неvirtуальных участников виртуальных турниров и для участников неvirtуальных турниров.
<code>stand_score_attr</code>	146	Атрибуты колонки «очки» таблицы текущих результатов (в режимах турнира <i>kirov</i> или <i>olympiad</i> ).
<code>stand_self_row_attr</code>	149	Атрибуты строки таблицы текущих результатов, соответствующей текущей команде при выводе текущих результатов данной команды в виртуальном турнире (только для виртуальных турниров).
<code>stand_solved_attr</code>	146	Атрибуты колонки «количество решённых задач» таблицы текущих результатов.
<code>stand_table_attr</code>	144	Атрибуты тега <code>&lt;table&gt;</code> таблицы текущих результатов.
<code>stand_team_attr</code>	145	Атрибуты колонки «команда» таблицы текущих результатов.
<code>stand_time_attr</code>	147	Атрибуты для вывода затраченного времени в ячейке таблицы текущих результатов (в режиме турнира <i>ACM</i> ).
<code>stand_u_row_attr</code>	150	Атрибуты строки таблицы текущих результатов для участников виртуального турнира с неопределённым статусом.
<code>stand_v_row_attr</code>	149	Атрибуты строки таблицы текущих результатов для виртуальных участников виртуальных турниров.
<code>status_dir</code>	100	Каталог файлов состояния программы <b>serve</b> .
<code>team_download_time</code>	129	Период скачивания участником своих решений.
<code>team_enable_ce_view</code>	154	Флаг разрешения просмотра участником протокола компиляции программы в случае ошибки компиляции.

*Продолжение на следующей странице*



Название	Стр.	Описание
<code>team_enable_rep_view</code>	127	Флаг разрешения просмотра участником протоколов тестирования.
<code>team_enable_src_view</code>	127	Флаг разрешения просмотра участником исходного текста решений.
<code>team_extra_dir</code>	155	Каталог для сохранения состояния клиента турнира.
<code>team_info_url</code>	126	Шаблон URL с информацией об участнике.
<code>team_page_quota</code>	157	Квота количества напечатанных страниц для участника турнира.
<code>team_show_judge_report</code>	154	Флаг доступа участников к судейскому протоколу тестирования.
<code>team_report_archive_dir</code>	99	Каталог архива протоколов тестирования, предназначенных для участников.
<code>test_dir</code>	94	Каталог с тестами к задачам.
<code>test_sfx</code>	115	Суффикс имён файлов с тестами.
<code>tests_to_accept</code>	130	Количество тестов для принятия задачи в режиме <i>OLYMPIAD</i> .
<code>tgz_dir</code>	141	Каталог с архивами рабочих каталогов для тестирования решений.
<code>tgz_sfx</code>	142	Суффиксы архивов рабочих каталогов.
<code>timelimit_sound</code>	118	Звук, проигрываемый при истечении лимита времени.
<code>user_priority_adjustment</code>	161	Спецификации изменения приоритета тестирования в зависимости от идентификатора пользователя.
<code>use_dir_hierarchy</code>	152	Флаг хранения архивных файлов в иерархической структуре каталогов.
<code>use_gzip</code>	153	Флаг сжатия архивных файлов.
<code>variant_map_file</code>	136	Путь к файлу с распределением вариантов между участниками.
<code>var_dir</code>	97	Каталог рабочих файлов турнира.
<code>virtual</code>	114	Флаг виртуального турнира.
<code>work_dir</code>	101	Рабочий каталог.
<code>wrong_sound</code>	118	Звук, проигрываемый при неправильном ответе тестируемой программы.

Текущая задача (секция <a href="#">problem</a> )	
Pi	Идентификатор задачи (поле <a href="#">id</a> )
Ps	Короткое имя задачи (поле <a href="#">short_name</a> )
Pl	Полное имя задачи (поле <a href="#">name</a> )
Текущий тестер (секция <a href="#">tester</a> )	
Ti	Идентификатор тестера (поле <a href="#">id</a> )
Tn	Имя тестера (поле <a href="#">name</a> )
Tj	Идентификатор задачи, которую проверяет тестер (поле <a href="#">problem</a> )
Tr	Короткое имя задачи, которую проверяет тестер (поле <a href="#">problem_name</a> )
Ta	Архитектура, на которой работает тестер (поле <a href="#">arch</a> )
Tk	Ключ тестера (поле <a href="#">key</a> )
Текущий участник	
Mi	Идентификатор участника
Mn	Имя участника
Ml	Регистрационное имя (login) участника
Mc	Поле «город» (city) регистрационной формы участника. Если данное поле не заполнялось, заменяется на пустую строку
MC	Поле «город (англ.)» (city_en) регистрационной формы участника
Mo	Поле «страна» (country) регистрационной формы участника
MO	Поле «страна (англ.)» (country_en) регистрационной формы участника
Mt	Поле «краткое название учебного заведения» (inst_short) регистрационной формы участника
MT	Поле «краткое название учебного заведения (англ.)» (inst_short_en) регистрационной формы участника
Mu	Поле «название учебного заведения» (inst) регистрационной формы участника
MU	Поле «название учебного заведения (англ.)» (inst_en) регистрационной формы участника

Таблица 2.9: Спецификаторы форматного преобразования

## 2.11.4 Глобальные параметры

В данном разделе рассматриваются глобальные конфигурационные переменные конфигурационного файла `serve.cfg`.

### Переменная `contest_id`

<b>Имя переменной:</b>	<code>contest_id</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используется:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>integer</i>
<b>Может отсутствовать:</b>	<i>нет</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт идентификатор обслуживаемого турнира. Идентификатор турнира — целое число больше нуля. Название турнира и информация общего характера о турнире берётся из конфигурационного файла турнира `contest.xml`.

#### Пример.

```
contest_id = 20
```

### Переменная `contests_dir`

<b>Имя переменной:</b>	<code>contests_dir</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используется:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>нет</i>
<b>Может повторяться:</b>	<i>нет</i>

Данная конфигурационная переменная задаёт путь к каталогу, в котором находятся файлы описания турниров `contest.xml`. Каждое описание турнира находится в отдельном файле с именем, равным шестизначному номеру турнира, например, `000000.xml`. Используемое описание турнира определяется с помощью конфигурационной переменной `contest_id`.

#### Пример.

```
contests_dir = "/var/ejudge/contests"
```

## Переменная `socket_path`

Имя переменной:	<code>socket_path</code>
Содержится в:	<code>global</code>
Используется:	<code>serve</code>
Тип содержимого:	<i>путь к UNIX-сокету</i>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт путь к UNIX-сокету, который используется для связи с сервером информации о пользователях `userlist-server`. Когда программа `serve` стартует, UNIX-сокет должен существовать и быть доступным для пользователя, от имени которого запускается программа.

**Пример.**

```
socket_path = "/var/ejudge/userlist-socket"
```

## Переменная `sleep_time`

Имя переменной:	<code>sleep_time</code>
Содержится в:	<code>global</code>
Используется:	<code>serve, compile, run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	1000
Может повторяться:	<i>нет</i>

Программа `serve` обменивается с программами `compile` и `run` посредством разделяемых файлов в общей файловой системе. Например, чтобы отправить полученное от участника решение на компиляцию, программа `serve` записывает служебный файл и файл решения в специальные каталоги обмена с программой `compile`. Все три программы периодически проверяют, не появились ли новые файлы в каталогах обмена, и в случае появления новых файлов выполняют соответствующие действия. Переменная `sleep_time` позволяет задать интервал времени в миллисекундах между опросами каталогов обмена для программ `serve`, `compile` и `run`. Для программы `serve` интервал времени может переопределяться с помощью переменной `serve_sleep_time`. Значение конфигурационной переменной `sleep_time` по умолчанию равно 1000 (миллисекунд), что соответствует задержке в 1 секунду.

## Переменная `serve_sleep_time`

Имя переменной:	<code>serve_sleep_time</code>
Содержится в:	<code>global</code>
Используется:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	равно значению <code>sleep_time</code>
Может повторяться:	<i>нет</i>

Данная переменная задаёт интервал времени между опросами каталогов обмена программы `serve`. Интервал времени задаётся в миллисекундах. Если конфигурационная переменная `serve_sleep_time` не установлена, используется значение конфигурационной переменной `sleep_time`, а если и эта переменная не установлена, используется значение по умолчанию 1000, что соответствует одной секунде задержки.

## Переменная `contest_time`

Имя переменной:	<code>contest_time</code>
Содержится в:	<code>global</code>
Используется:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

Данная конфигурационная переменная задаёт время турнира в секундах. Время в секундах должно быть целым значением, больше или равно 0. Если время равно 0, продолжительность турнира **не ограничена**. В этом случае параметры `board_fog_time`, `board_unfog_time` не используются. Турнир продолжается до тех пор, пока не будет остановлен администратором.

### Пример.

```
contest_time = 0
```

Здесь задаётся неограниченная продолжительность турнира. Обратите внимание, что в этом случае следует изменить значения конфигурационных переменных **TODO** — каких?

```
contest_time = 300
```

В данном случае продолжительность турнира устанавливается равной 5 часам.

## Переменная `board_fog_time`

Имя переменной:	<code>board_fog_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>60</code>
Может повторяться:	<code>нет</code>

**Описание.** По регламенту проведения командных соревнований по программированию АСМ таблица текущих результатов «замораживается», то есть перестаёт обновляться за 60 минут до конца соревнования. С помощью переменной `board_fog_time` можно установить любое другое время заморозки таблицы результатов. Время задаётся в минутах, то есть, например, значение 60 означает заморозку таблицы результатов за час до конца. Значение переменной должно быть больше нуля. Значение 0 означает отсутствие заморозки, то есть текущие результаты турниру будут обновляться до самого последнего момента. Заморозка действует только на автоматические обновления таблицы результатов. Ручное обновление таблицы возможно вне зависимости от времени заморозки (с помощью CGI-программы `master`). Если после того, как заморозка вступила в действие, турнир был продлён так, что согласно новой продолжительности турнира заморозка в действие не вступила, таблица результатов будет обновлена, как только поступит новое решение или по команде обновления от администратора турнира.

Данная конфигурационная переменная имеет синонимичную ей глобальную конфигурационную переменную `stand_freeze_time`. Эти две конфигурационные переменные являются синонимами и взаимозаменяемы.

**Пример.** Приведённый ниже пример включает заморозку результатов за 30 минут до конца турнира.

```
board_fog_time = 30
```

## Переменная `stand_freeze_time`

Имя переменной:	<code>stand_freze_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>60</code>
Может повторяться:	<code>нет</code>
Версия <code>ejudge</code> :	начиная с 2.1.24

**Описание.** Данная конфигурационная переменная является синонимом конфигурационной переменной `board_fog_time`. Для дальнейшей информации обратитесь к описанию этой переменной.

## Переменная `board_unfog_time`

Имя переменной:	<code>board_unfog_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>120</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт время в минутах после окончания турнира, после которого снова возможны автоматические обновления таблицы результатов. Ручное обновление таблицы результатов после окончания турнира возможно вне зависимости от значения этой конфигурационной переменной. Значение переменной должно быть больше или равно 0. Если указано значение 0, автоматическое обновление результатов возможно сразу же после окончания турнира.

Данная конфигурационная переменная имеет синонимичную ей глобальную конфигурационную переменную `stand_melt_time`. Эти две конфигурационные переменные являются синонимами и взаимозаменяемы.

**Пример.** Приведённый ниже пример отключает заморозку результатов после окончания турнира.

```
board_unfog_time = 0
```

## Переменная `stand_melt_time`

Имя переменной:	<code>stand_melt_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>60</code>
Может повторяться:	<code>нет</code>
Версия <b>ejudge</b> :	начиная с 2.1.24

**Описание.** Данная конфигурационная переменная является синонимом конфигурационной переменной `board_unfog_time`. Для дальнейшей информации обратитесь к описанию этой переменной.

## Переменная `max_run_size`

Имя переменной:	<code>max_run_size</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>65535</code>
Может повторяться:	<code>nem</code>

Данная конфигурационная переменная устанавливает максимальный размер программы, принимаемой на проверку. Если размер программы превышает лимит, программа к проверке не принимается, а участнику выводится сообщение об ошибке. При увеличении максимального размера программы учтите, что действует ограничение размера данных, принимаемых от клиента, равное 128 килобайт. Чтобы изменить это ограничение исправьте значение константы `MAX_VALUE_SIZE` в исходном файле `cgi.c` и перекомпилируйте систему `ejudge`.

**Пример.** Данный пример устанавливает максимальный размер программы равным 100 килобайт.

```
max_run_size = 102400
```

## Переменная `max_run_total`

Имя переменной:	<code>max_run_total</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>2097152</code>
Может повторяться:	<code>nem</code>

Данная конфигурационная переменная устанавливает максимальный суммарный размер всех решений от одного участника (квота размера). Если размер очередной присланной на проверку программы таков, что в сумме с предыдущими программами того же участника он превышает квоту, программа на проверку не принимается, а участнику выдаётся сообщение об ошибке. Размер квоты по умолчанию — 2 мегабайта, чего более чем достаточно для обычного использования.



## Переменная `max_run_num`

Имя переменной:	<code>max_run_num</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>200</code>
Может повторяться:	<code>нет</code>

Данная конфигурационная переменная устанавливает максимальное количество присылаемых на проверку программ от одного участника. Количество присланных программ суммируется для каждого участника по всем задачам. Если максимальное количество присылаемых программ превышено, очередная программа будет отвергнута, а участнику будет выдано сообщение об ошибке. Значение по умолчанию — 200, чего должно быть достаточно для разумного использования.

## Переменная `max_clar_size`

Имя переменной:	<code>max_clar_size</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>1024</code>
Может повторяться:	<code>нет</code>

Данная конфигурационная переменная устанавливает максимальный размер сообщения или вопроса от участника. Если размер текста вопроса превышает лимит, вопрос не принимается, а участнику выводится сообщение об ошибке. При увеличении максимального размера сообщения учтите, что действует ограничение размера данных, принимаемых от клиента, равное 128 килобайт. Чтобы изменить это ограничение исправьте значение константы `MAX_VALUE_SIZE` в исходном файле `cgi.c` и перекомпилируйте систему `ejudge`.

**Пример.** Данный пример устанавливает максимальный размер сообщения равным 2 килобайта.

```
max_clar_size = 2048
```

## Переменная `max_clar_total`

<b>Имя переменной:</b>	<code>max_clar_total</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<code>integer</code>
<b>Может отсутствовать:</b>	<code>da</code>
<b>Значение по умолчанию:</b>	<code>40960</code>
<b>Может повторяться:</b>	<code>нет</code>

Данная конфигурационная переменная устанавливает максимальный суммарный размер текста всех сообщений от одного участника (квота размера). Если размер очередного сообщения таков, что в сумме с предыдущими сообщениями того же участника он превышает квоту, сообщение не принимается, а участнику выдаётся сообщение об ошибке. Размер квоты по умолчанию — 40 килобайт, чего более чем достаточно для обычного использования.

## Переменная `max_clar_num`

<b>Имя переменной:</b>	<code>max_clar_num</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<code>integer</code>
<b>Может отсутствовать:</b>	<code>da</code>
<b>Значение по умолчанию:</b>	<code>50</code>
<b>Может повторяться:</b>	<code>нет</code>

Данная конфигурационная переменная устанавливает максимальное количество сообщений от одного участника. Если максимальное количество сообщений превышено, очередное сообщение будет отвергнуто, а участнику будет выдано сообщение об ошибке. Значение по умолчанию — 50, чего должно быть достаточно для разумного использования.

## Переменная `charset`

Имя переменной:	<b>charset</b>
Содержится в:	<a href="#">global</a>
Используются:	<b>serve</b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	<code>iso8859-1</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт кодировку, в которой генерируются все html-страницы, передаваемые клиентам **team**, **master**, **judge**. В этой кодировке также сохраняются базы дампы баз данных.

**Внимание.** В текущей версии системы **ejudge** поддержка разных кодировок реализована неудовлетворительно. Не гарантируется корректная работа системы, если кодировки, заданные в параметрах `charset` конфигурационных файлов CGI-программ **team**, **master**, **judge**, и кодировка, заданная в этом файле, не совпадают. Кроме того, в действительности поддерживаются только две кодировки: `iso8859-1` и `koi8-r`. Корректная работа с другими кодировками также не гарантируется.

### Пример.

```
charset = koi8-r
```

## Переменная `standings_charset`

Имя переменной:	<b>standings_charset</b>
Содержится в:	<a href="#">global</a>
Используются:	<b>serve</b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	совпадает со значением <code>charset</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт кодировку, в которой генерируется статическая таблица результатов турнира. Статическая таблица результатов — это html-файл, генерируемый в каталоге файлов состояния сервера турнира, который обновляется после каждой посылки участника. Назначение этого файла — дать возможность всем желающим просматривать текущие результаты турнира. По умолчанию предполагается, что кодировка этого файла совпадает с глобальной кодировкой сервера турнира (см. конфигурационную переменную [charset](#)). Если переменная `standings_charset` задаёт другую кодировку, выполняется перекодирование файла.

**Внимание.** Корректная работа системы **ejudge** в случае, если переменная `standings_charset` задаёт кодировку, отличную от `cp1251`, не гарантируется.

### Пример.

```
charset = cp1251
```

## Переменная `enable_l10n`

Имя переменной:	<code>enable_l10n</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная включает локализацию сообщений во фрагментах html-файлов, генерируемых программой `serve` для CGI-программ `team`, `judge`, `master`. Если локализация включена, требуемый язык сообщений определяется клиентом. Чтобы включить локализацию, достаточно написать в конфигурационном файле имя переменной `enable_l10n`.

**Замечание.** В настоящее время только CGI-программа `team` поддерживает локализацию сообщений. CGI-программы `judge` и `master` выводят сообщения только на английском языке. В настоящее время поддерживаются два языка: английский и русский (в кодировке `koï8-r`). Поскольку для локализации используются стандартные средства локализации (функция `gettext(3)`, доступная на Linux), добавление новых языков является достаточно простой задачей, если существующие проблемы с кодировками будут решены.

## Переменная `l10n_dir`

Имя переменной:	<code>l10n_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	(пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Переводы сообщений находятся не в исполняемом файле программы, а в отдельном каталоге. Чтобы сервер турнира `serve` мог использовать их, необходимо указать полный путь к каталогу сообщений (message catalog), в котором находятся файлы с переведёнными сообщениями. По умолчанию после выполнения команды `make install` каталоги сообщений помещаются в каталог, определяемой переменной `INST_LOCALE_PATH` в `makefile`. Если каталог не указан, или указан неправильно, локализация сообщений работать не будет, даже если параметр `enable_l10n` установлен в *true*.

### Пример.

```
enable_l10n
l10n_dir = "/usr/share/locale"
```

## Переменная `standings_locale`

Имя переменной:	<code>standings_locale</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>строка</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	(пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет локализовать статическую таблицу результатов турнира. По умолчанию все элементы таблицы выводятся на английском языке (например, “Place” в заголовке столбца таблицы). Установка значения данной конфигурационной переменной в имя другого локального окружения (locale) позволяет изменять язык. Например, если включено русское локальное окружение `ru_RU.KOI8-R`, тот же заголовок столбца таблицы будет называться «Место». Перекодировка таблицы, управляемая конфигурационной переменной `standings_charset`, выполняется после генерации таблицы результатов.

**Внимание.** В настоящее время поддерживается только русское локальное окружение `ru_RU.KOI8-R`.

**Пример.**

```
standings_locale = "ru_RU.KOI8-R"
```

## Переменная `root_dir`

Имя переменной:	<code>root_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve, compile, run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет задать путь к каталогу, в котором находятся конфигурационные и рабочие файлы системы **ejudge**. В подкаталогах этого каталога рекомендуется размещать конфигурационные файлы `serve`, тесты, правильные ответы, проверяющие программы. Кроме того, в подкаталоге `var` этого каталога будут размещаться рабочие файлы турнира: архив посылок участников и их вопросов, архив протоколов запуска и временные файлы. Переменная должна быть обязательно задана в конфигурационном файле.

**Пример.**

```
root_dir = /home/judges/20
```

## Переменная `conf_dir`

<b>Имя переменной:</b>	<b><code>conf_dir</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve, compile, run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>conf</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет задать каталог, в котором находятся конфигурационные файлы, тесты, проверяющие программы, ответы к тестам. Если эта переменная не задана, конфигурационным каталогом является подкаталог `conf` каталога, заданного в переменной `root_dir`. Если конфигурационная переменная `conf_dir` задана, полный путь к каталогу конфигурационных файлов образуется конкатенацией пути, указанного в переменной `root_dir` и пути, указанного в переменной `conf_dir`.

## Переменная `script_dir`

<b>Имя переменной:</b>	<b><code>script_dir</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>compile, run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>scripts</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная определяет путь к каталогу скриптов компиляции и запуска программ. В этом каталоге находятся программы (обычно это скрипты на языке `bash`), вызываемые для компиляции посылок участников, а также вспомогательные программы, назначение которых — запустить проверяемое решение участника. Полный путь к каталогу скриптов определяется по следующим правилам:

- Если значение переменной не задано, используется значение `scripts`.
- Если значение переменной начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `script_dir`.

### Пример.

```
script_dir = /usr/share/ejudge/scripts
```

## Переменная `test_dir`

Имя переменной:	<code>test_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>tests</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранятся тесты для всех задач данного турнира. Сами тесты находятся в подкаталогах этого каталога. Тесты для некоторой задачи с кратким именем `A` (задаваемом конфигурационной переменной `short_name`) находятся в подкаталоге `A` тестового каталога. Полный путь к каталогу с тестами определяется по следующим правилам:

- Если значение переменной `test_dir` не задано, используется значение `tests`.
- Если значение переменной `test_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `test_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `test_dir`.

**Пример.** В следующем примере путь к каталогу тестов устанавливается в `${root_dir}/conf/../tests`, где `${root_dir}` — значение конфигурационной переменной `root_dir`. Этот путь эквивалентен пути `${root_dir}/tests`.

```
script_dir = ../tests
```

## Переменная `corr_dir`

Имя переменной:	<code>corr_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>correct</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранятся правильные ответы к тестам для всех задач данного турнира. Сами правильные ответы находятся в подкаталогах этого каталога. Правильные ответы для некоторой задачи с кратким именем А (задаваемом конфигурационной переменной `short_name`) находятся в подкаталоге А каталога правильных ответов. Полный путь к каталогу с правильными ответами определяется по следующим правилам:

- Если значение переменной `corr_dir` не задано, используется значение `correct`.
- Если значение переменной `corr_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `corr_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `corr_dir`.

**Пример.** В следующем примере путь к каталогу тестов устанавливается в `${root_dir}/conf/../tests`, где `${root_dir}` — значение конфигурационной переменной `root_dir`. Этот путь эквивалентен пути `${root_dir}/tests`. Таким образом, тесты к задачам и правильные ответы к тестам находятся в одном каталоге.

```
script_dir = ../tests
```



## Переменная `checker_dir`

<b>Имя переменной:</b>	<code>checker_dir</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>checkers</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранятся проверяющие программы для всех задач данного турнира. Полный путь к проверяющей программе является конкатенацией имени каталога, определённого в данной конфигурационной переменной, и имени самой программы, определённого в конфигурационной переменной `check_cmd`. Полный путь к каталогу с проверяющими программами определяется по следующим правилам:

- Если значение переменной `checker_dir` не задано, используется значение `checkers`.
- Если значение переменной `checker_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `checker_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `checker_dir`.

**Пример.** В следующем примере путь к каталогу тестов устанавливается в `${root_dir}/conf/../checkers`, где `${root_dir}` — значение конфигурационной переменной `root_dir`. Этот путь эквивалентен пути `${root_dir}/checkers`.

```
script_dir = ../checkers
```

## Переменная `var_dir`

<b>Имя переменной:</b>	<b><code>var_dir</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>var</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранятся архивные и рабочие файлы турнира. Если переменная в конфигурационном файле не установлена, её значение полагается равным `var`. Полный путь к каталогу получается конкатенацией значения переменной `root_dir` и данной переменной.

**Внимание.** Изменять значение по умолчанию данной переменной не рекомендуется. Изменение должно быть согласовано с конфигурационными файлами CGI-программ.

## Переменная `archive_dir`

<b>Имя переменной:</b>	<b><code>archive_dir</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>archive</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором размещаются архивы турнира. Полный путь к каталогу с проверяющими программами определяется по следующим правилам:

- Если значение переменной `archive_dir` не задано, используется значение `archive`.
- Если значение переменной `archive_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `archive_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `archive_dir`.

## Переменная `clar_archive_dir`

Имя переменной:	<code>clar_archive_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>clars</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором размещается архив сообщений турнира. Полный путь к каталогу с архивом сообщений определяется по следующим правилам:

- Если значение переменной `clar_archive_dir` не задано, используется значение `clars`.
- Если значение переменной `clar_archive_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `clar_archive_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `archive_dir` и значения переменной `clar_archive_dir`.

## Переменная `run_archive_dir`

Имя переменной:	<code>run_archive_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>runs</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором размещается архив посылок от участников турнира. Полный путь к каталогу с архивом посылок определяется по следующим правилам:

- Если значение переменной `run_archive_dir` не задано, используется значение `runs`.
- Если значение переменной `run_archive_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `run_archive_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `archive_dir` и значения переменной `run_archive_dir`.

## Переменная `report_archive_dir`

Имя переменной:	<code>report_archive_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>reports</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором размещается архив протоколов запусков решений участников турнира. Полный путь к каталогу с архивом протоколов определяется по следующим правилам:

- Если значение переменной `report_archive_dir` не задано, используется значение `reports`.
- Если значение переменной `report_archive_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `report_archive_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `archive_dir` и значения переменной `report_archive_dir`.

## Переменная `team_report_archive_dir`

Имя переменной:	<code>team_report_archive_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>teamreports</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором размещается архив протоколов участников запусков решений участников турнира. Полный путь к каталогу с архивом протоколов определяется по следующим правилам:

- Если значение переменной `team_report_archive_dir` не задано, используется значение `teamreports`.
- Если значение переменной `team_report_archive_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `team_report_archive_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `archive_dir` и значения переменной `team_report_archive_dir`.

## Переменная `status_dir`

Имя переменной:	<code>status_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>status</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором находятся файлы состояния турнира. К таким файлам относятся таблица текущих результатов и файл статуса сервера. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `status_dir` не задано, используется значение `status`.
- Если значение переменной `status_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `status_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `status_dir`.

**Внимание.** Изменять значение по умолчанию данной переменной не рекомендуется. Изменение должно быть согласовано с конфигурационными файлами CGI-программ.

## Переменная `work_dir`

Имя переменной:	<code>work_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>compile</code> , <code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>work</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, используемому для временных файлов программами `compile` и `run`. Каждая из этих программ использует свой подкаталог в этом каталоге. Например, программа `compile` копирует в рабочий каталог файл исходного текста, компилирует его в исполняемый файл в рабочем каталоге, который затем копируется в каталоги обмена с программой `serve`. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `work_dir` не задано, используется значение `work`.
- Если значение переменной `work_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `work_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `work_dir`.

**Внимание.** Если планируется параллельная работа нескольких программ `compile` или нескольких программ `run` на нескольких компьютерах с обменом через каталоги, монтируемые с сервера, один и тот же рабочий каталог компиляции и запуска программ не должен разделяться несколькими работающими программами.

## Переменная `compile_work_dir`

Имя переменной:	<code>compile_work_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>compile</code> , <code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>compile</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, используемому для временных файлов программой `compile`. Программа копирует в рабочий каталог файл исходного текста, компилирует его в исполняемый файл в рабочем каталоге, который затем копируется в каталоги обмена с программой `serve`. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `compile_work_dir` не задано, используется значение `compile`.
- Если значение переменной `compile_work_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `compile_work_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `work_dir` и значения переменной `compile_work_dir`.

**Внимание.** Если планируется параллельная работа нескольких программ `compile` на нескольких компьютерах с обменом через каталоги, монтируемые с сервера, один и тот же рабочий каталог компиляции программ не должен разделяться несколькими работающими программами. Наилучший способ добиться этого — расположить рабочий каталог на локальном диске.

### Пример.

```
compile_work_dir = /tmp/compile
```

## Переменная `run_work_dir`

<b>Имя переменной:</b>	<code>run_work_dir</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>compile, run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>runwork</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, используемому для временных файлов программой `run`. В этом каталоге хранится временная копия тестируемой программы, а также файлы протокола запуска тестируемой программы. Тестируемые программы запускаются в другом каталоге, задаваемом конфигурационной переменной `run_check_dir`. После завершения проверки программы рабочий каталог очищается. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `run_work_dir` не задано, используется значение `runwork`.
- Если значение переменной `run_work_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `run_work_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `work_dir` и значения переменной `run_work_dir`.

**Внимание.** Если планируется параллельная работа нескольких программ `run` на нескольких компьютерах с обменом через каталоги, монтируемые с сервера, один и тот же рабочий каталог запуска программ не должен разделяться несколькими работающими программами. Наилучший способ добиться этого — расположить рабочий каталог на локальном диске.

### Пример.

```
run_work_dir = /tmp/runwork
```



## Переменная `run_check_dir`

<b>Имя переменной:</b>	<code>run_check_dir</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>runcheck</code>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, используемому для тестирования программой `run`. Для запуска проверяемой программы на каждом тесте в этот каталог копируется сама программа, затем входной файл теста. В каталоге создаются пустые выходные файлы. После этого запускается тестируемая программа. После каждого запуска и проверки результатов содержимое каталога очищается. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `run_check_dir` не задано, используется значение `runcheck`.
- Если значение переменной `run_check_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `run_check_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `work_dir` и значения переменной `run_check_dir`.

**Внимание.** Если планируется параллельная работа нескольких программ `run` на нескольких компьютерах с обменом через каталоги, монтируемые с сервера, один и тот же каталог запуска программ не должен разделяться несколькими работающими программами. Наилучший способ добиться этого — расположить рабочий каталог на локальном диске.

### Пример.

```
run_check_dir = /tmp/runcheck
```

## Переменная `compile_dir`

Имя переменной:	<code>compile_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code> , <code>compile</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>compile</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная переменная задаёт каталоги обмена между программами `serve` и `compile`. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `compile_dir` не задано, используется значение `compile`.
- Если значение переменной `compile_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `compile_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `compile_dir`.

Каталог обмена содержит в себе несколько подкаталогов, назначение которых описывается ниже.

- Подкаталог **queue** используется для передачи информации от программы `serve` к программе `compile`. Программа `serve` помещает в этот каталог небольшие файлы, содержащие служебную информацию о программе, которую необходимо скомпилировать (так называемый «пакет задания компиляции»). Программа `compile` периодически просматривает этот каталог и при появлении в нём новых пакетов задания компиляции выполняет их. Этот каталог может использоваться совместно несколькими программами `serve`, обслуживающими разные турниры, и несколькими программами `compile`, например, работающими на разных компьютерах в сети. Этот каталог имеет специальную структуру, чтобы предотвратить синхронизационные ошибки при одновременном доступе нескольких программ.
- Подкаталог **src** используется для передачи информации от программы `serve` к программе `compile`. Программа `serve` помещает в этот каталог файл исходного текста программы, которую необходимо скомпилировать. Файл с текстом программы имеет то же самое имя, что и пакет задания компиляции для этой программы. Программа `compile` считывает файл исходного текста после файла пакета задания компиляции. Этот каталог может использоваться совместно несколькими программами `serve`, обслуживающими разные турниры, и несколькими программами `compile`. Синхронизация доступа поддерживается в каталоге `queue`, поэтому никакой дополнительной синхронизации в каталоге `src` не требуется.
- Подкаталог **status** используется для передачи информации от программы `compile` к программе `serve`. В этот каталог программа `compile` помещает небольшой файл с информацией о результате компиляции (так называемый «пакет результата компиляции»). Программа `serve` периодически просматривает данный каталог и при появлении в нём

новых файлов обновляет своё внутреннее состояние. Поскольку каталог может использоваться одновременно несколькими программами **compile** и одной программой **serve**, он имеет специальную структуру для предотвращения синхронизационных ошибок.

- Подкаталог **report** используется для передачи информации от программы **compile** к программе **serve**. В этот каталог программа **compile** помещает список ошибок, выданных при неуспешной компиляции, или скомпилированный исполняемый файл в случае успешной компиляции. Каталог может использоваться одновременно несколькими программами **compile** и одной программой **serve**, но синхронизация доступа ведётся с помощью каталога **status**, поэтому никакой дополнительной синхронизации в каталоге **report** не требуется.
- Символическая ссылка **<номер>**, где **<номер>** — четырёхзначный идентификатор турнира. Эта символическая ссылка создаётся в каталоге, задаваемом переменной **compile\_dir**, программы **compile**, и указывает на каталог, задаваемый переменной **compile\_dir**, программы **serve**. Если эти каталоги совпадают, символическая ссылка указывает на каталог, в котором она находится. С помощью этой символической ссылки программа **compile** может обслуживать одновременно несколько турниров, записывая результаты компиляции в каталог обмена только соответствующего турнира.

**Пример.** С помощью задания каталога обмена можно добиться того, что несколько одновременно работающих серверов турнира будут использовать одну программу компиляции **compile**. Для этого создаётся отдельный каталог для программы **compile**, например **/var/ejudge/compile**. Этот каталог указывается в качестве коревого в конфигурационном файле программы **compile** с помощью строки

```
root_dir = /var/ejudge/compile
```

Программа **compile** запускается в каталоге **/var/ejudge/compile**. Теперь во всех конфигурационных файлах серверов турниров для использования этой программы компиляции достаточно установить переменную **compile\_dir** в следующее значение:

```
compile_dir = /var/ejudge/compile/var/compile
```

## Переменная `run_dir`

Имя переменной:	<code>run_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>run</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная переменная задаёт каталоги обмена между программами `serve` и `run`. Полный путь к этому каталогу определяется по следующим правилам:

- Если значение переменной `run_dir` не задано, используется значение `run`.
- Если значение переменной `run_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `run_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `run_dir`.

Каталог обмена содержит в себе несколько подкаталогов, назначение которых описывается ниже.

- Подкаталог **`queue`** используется для передачи информации от программы `serve` к программе `run`. Программа `serve` помещает в этот каталог небольшие файлы, содержащие служебную информацию о программе, которую необходимо протестировать (так называемый «пакет задания тестирования»). Программа `run` периодически просматривает этот каталог и при появлении в нём новых пакетов задания тестирования выполняет их. Этот каталог может использоваться совместно несколькими программами `serve`, обслуживающими разные турниры, и несколькими программами `run`, например, работающими на разных компьютерах в сети. Этот каталог имеет специальную структуру, чтобы предотвратить синхронизационные ошибки при одновременном доступе нескольких программ.
- Подкаталог **`exe`** используется для передачи информации от программы `serve` к программе `run`. Программа `serve` помещает в этот каталог исполняемый файл программы, которую необходимо протестировать. Исполняемый файл программы имеет то же самое имя, что и пакет задания тестирования для этой программы. Программа `run` считывает исполняемый файл после файла пакета задания компиляции. Этот каталог может использоваться совместно несколькими программами `serve`, обслуживающими разные турниры, и несколькими программами `run`. Синхронизация доступа поддерживается в каталоге `queue`, поэтому никакой дополнительной синхронизации в каталоге `exe` не требуется.
- Подкаталог **`status`** используется для передачи информации от программы `run` к программе `serve`. В этот каталог программа `run` помещает небольшой файл с информацией о результате тестирования (так называемый «пакет результата тестирования»). Программа `serve` периодически просматривает данный каталог и при появлении в нём

новых файлов обновляет своё внутреннее состояние. Поскольку каталог может использоваться одновременно несколькими программами `run` и одной программой `serve`, он имеет специальную структуру для предотвращения синхронизационных ошибок.

- Подкаталог **report** используется для передачи информации от программы `run` к программе `serve`. В этот каталог программа `run` помещает судейский вариант протокола тестирования. Каталог может использоваться одновременно несколькими программами `run` и одной программой `serve`, но синхронизация доступа ведётся с помощью каталога `status`, поэтому никакой дополнительной синхронизации в каталоге `report` не требуется.
- Подкаталог **teamreport** используется для передачи информации от программы `run` к программе `serve`. В этот каталог программа `run` помещает пользовательский вариант протокола тестирования, если эта опция включена в конфигурационном файле турнира. Каталог может использоваться одновременно несколькими программами `run` и одной программой `serve`, но синхронизация доступа ведётся с помощью каталога `status`, поэтому никакой дополнительной синхронизации в каталоге `teamreport` не требуется.
- Символическая ссылка **<номер>**, где `<номер>` — четырёхзначный идентификатор турнира. Эта символическая ссылка создаётся в каталоге, задаваемом переменной `run_dir`, программы `run`, и указывает на каталог, задаваемый переменной `compile_dir`, программы `serve`. Если эти каталоги совпадают, символическая ссылка указывает на каталог, в котором она находится. С помощью этой символической ссылки программа `run` может обслуживать одновременно несколько турниров, записывая результаты тестирования в каталог обмена только соответствующего турнира.

**Пример.** С помощью задания каталога обмена можно добиться того, что несколько одновременно работающих серверов турнира будут использовать одну программу тестирования `run`. Для этого создаётся отдельный каталог для программы `run`, например `/var/ejudge/run`. Этот каталог указывается в качестве корневого в конфигурационном файле программы `run` с помощью строки

```
root_dir = /var/ejudge/run
```

Программа `run` запускается в каталоге `/var/ejudge/run`. Теперь во всех конфигурационных файлах серверов турниров для использования этой программы тестирования достаточно установить переменную `run_dir` в следующее значение:

```
run_dir = /var/ejudge/run/var/compile
```

## Переменная `serve_socket`

Имя переменной:	<code>serve_socket</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к UNIX-сокету</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>serve</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к UNIX-сокету, который используется программой `serve` для взаимодействия с CGI-программами. Программа `serve` создаёт UNIX-сокет при запуске и удаляет при завершении, если только в командной строке не был задан ключ `-S`. Полный путь к сокету определяется по следующим правилам.

- Если значение переменной `serve_socket` не задано, используется значение `serve`.
- Если значение переменной `serve_socket` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `serve_socket` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `var_dir` и значения переменной `serve_socket`.

**Внимание.** Изменять значение данной конфигурационной переменной не рекомендуется.

## Переменная `run_log_file`

Имя переменной:	<code>run_log_file</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>run.log</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к файлу, в котором программа `serve` хранит базу посылок пользователей. Полный путь к файлу определяется по следующим правилам.

- Если значение переменной `run_log_file` не задано, используется значение `run.log`.
- Если значение переменной `run_log_file` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к файлу, этот путь используется без изменений.
- Если значение переменной `run_log_file` не начинается с символа `'/'`, полный путь к файлу образуется конкатенацией значения переменной `var_dir` и значения переменной `run_log_file`.

**Внимание.** Изменять значение данной конфигурационной переменной не рекомендуется.

## Переменная `clar_log_file`

Имя переменной:	<code>clar_log_file</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>clar.log</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к файлу, в котором программа `serve` хранит базу сообщений. Полный путь к файлу определяется по следующим правилам.

- Если значение переменной `clar_log_file` не задано, используется значение `clar.log`.
- Если значение переменной `clar_log_file` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к файлу, этот путь используется без изменений.
- Если значение переменной `clar_log_file` не начинается с символа `'/'`, полный путь к файлу образуется конкатенацией значения переменной `var_dir` и значения переменной `clar_log_file`.

**Внимание.** Изменять значение данной конфигурационной переменной не рекомендуется.

## Переменная `cr_serialization_key`

Имя переменной:	<code>cr_serialization_key</code>
Содержится в:	<code>global</code>
Используются:	<code>compile</code> , <code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет установить идентификатор SYSV IPC семафора, который будет использоваться для сериализации работы программ `compile` и `run`. Если эта конфигурационная переменная не установлена, сериализация запуска проводится не будет, то есть, если несколько программ `compile` и (или) несколько программ `run` работают на одной машине, возможен случай, когда все программы будут работать одновременно. Если сериализация включена, то не более одной программы будет выполняться в каждый момент времени.

**Пример.**

```
cr_serialization_key = 12345
```

## Переменная `inactivity_timeout`

Имя переменной:	<code>inactivity_timeout</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	120
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная действует, если программа `serve` запущена с ключом `-S` в командной строке. Данный ключ передаёт программе `serve` файловый дескриптор UNIX-сокета, который будет использоваться как сокет команд программы `serve`. Этот аргумент командной строки используется, когда сервер турнира запускается по требованию программой `super-serve`. Конфигурационная переменная `inactivity_timeout` задаёт время в секундах, после которого программа `serve` завершает свою работу, если за указанный промежуток времени к ней не было ни одного обращения. Если значение конфигурационной переменной не установлено, предполагается значение 120 секунд (2 минуты). Если значение переменной установлено в 0, значение тайм-аута полагается неограниченным.

Для программы `run` данная конфигурационная переменная действует, если эта программа запущена с ключом `-S` в командной строке. Этот аргумент командной строки используется, когда сервер проверки решений запускается по требованию программой `super-serve`. Конфигурационная переменная `inactivity_timeout` задаёт время в секундах, после которого программа `run` завершает свою работу, если за указанный промежуток времени к ней не было ни одного обращения. Если значение конфигурационной переменной не установлено, предполагается значение 120 секунд (2 минуты). Если значение переменной установлено в 0, значение тайм-аута полагается неограниченным.

### Пример.

```
inactivity_timeout = 300
```



## Переменная `score_system`

Имя переменной:	<code>score_system</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>score_system</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>ACM</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная переменная определяет тип турнира. Поддерживаются три типа турниров, описываемых ниже.

- Тип турнира *ACM*. Этот тип задаётся строкой

```
score_system = acm
```

в конфигурационном файле турнира. Данный тип турнира соответствует регламенту проведения чемпионата мира среди студенческих команд ACM. Он имеет следующие характеристики:

1. Проверка решений ведётся непосредственно по ходу турнира. На каждую свою посылку участники немедленно получают ответ с результатами её тестирования.
  2. Участники ранжируются по числу решённых задач. Чем больше решено задач, тем выше место участника. Среди участников, решивших одинаковое количество задач, участники ранжируются по количеству штрафных очков. Чем меньше штрафных очков, тем выше место участника.
  3. Участник считается решившим задачу, если решение этой задачи, посланное участником, успешно прошло все тесты. Частичные решения не засчитываются.
  4. Штрафные очки для участника считаются как сумма штрафных очков по всем задачам, решённым участником. Нерешённые задачи не дают вклада в штрафные очки. Для решённой задачи штрафные очки по ней вычисляются как количество минут, прошедших от начала турнира до момента сдачи задачи, плюс по 20 минут за каждую предыдущую неуспешную попытку сдачи задачи. Попытки сдачи программы после успешной не учитываются.
- Тип турнира *KIROV*. Данный тип турниров придуман В. Матюхиным. Этот тип задаётся строкой

```
score_system = kirov
```

в конфигурационном файле турнира. Он имеет следующие характеристики:

1. Проверка решений ведётся непосредственно по ходу турнира. На каждую свою посылку участники немедленно получают ответ с результатами её тестирования.
2. Участники ранжируются по числу набранных баллов. Чем больше баллов набрал участник, тем выше его место.

3. Баллы, набранные участником, считаются как сумма баллов по каждой из задач турнира. Количество баллов, полученных участником за задачу, вычисляется как максимальное количество баллов, полученных участником за все попытки сдачи задачи.
4. Баллы, полученные участником за попытку вычисляются следующим образом:

- Если решение участника прошло все тесты, оно получает полный балл за эту задачу (см. конфигурационную переменную `full_score`) за вычетом произведения штрафных баллов за попытку (см. конфигурационную переменную `run_penalty`) помноженных на количество предыдущих попыток сдачи решения. Если в результате получается число, меньшее нуля, количество баллов полагается равным нулю.
- Если решение участника прошло часть тестов (либо не прошло ни одного теста), оно получает частичный балл за эту задачу в зависимости от конфигурационных переменных `test_score`, `test_score_list`, `test_sets` за вычетом произведения штрафных баллов за попытку (см. конфигурационную переменную `run_penalty`) помноженных на количество предыдущих попыток сдачи решения. Если в результате получается число, меньшее нуля, количество баллов полагается равным нулю.

- Тип турнира *OLYMPIAD*. Этот тип задаётся строкой

```
score_system = olympiad
```

в конфигурационном файле турнира. Данный тип турнира соответствует регламенту проведения российских и международных олимпиад по информатике. Он имеет следующие характеристики.

1. Полная проверка решений участников ведётся после окончания основного времени турнира. Во время турнира присылаемые решения проверяются на нескольких первых тестах (см. конфигурационную переменную `tests_to_accept`). Если решение участника прошло все предварительные тесты, оно принимается для последующей проверки, а если решение не прошло хотя бы один тест, оно для проверки не принимается и в окончательной проверке решений по окончании турнира не участвует.
2. Участники ранжируются по числу набранных баллов. Чем больше баллов набрал участник, тем выше его место.
3. Баллы, набранные участником, считаются как сумма баллов по каждой из задач турнира. Количество баллов, полученных участником за задачу, вычисляется как максимальное количество баллов, полученных участником за все попытки сдачи задачи.
4. Баллы, полученные участником за попытку вычисляются следующим образом:
  - Если решение участника прошло все тесты, оно получает полный балл за эту задачу (см. конфигурационную переменную `full_score`) за вычетом произведения штрафных баллов за попытку (см. конфигурационную переменную `run_penalty`) помноженных на количество предыдущих попыток сдачи решения. Если в результате получается число, меньшее нуля, количество баллов полагается равным нулю.

- Если решение участника прошло часть тестов (либо не прошло ни одного теста), оно получает частичный балл за эту задачу в зависимости от конфигурационных переменных `test_score`, `test_score_list`, `test_sets` за вычетом произведения штрафных баллов за попытку (см. конфигурационную переменную `run_penalty`) помноженных на количество предыдущих попыток сдачи решения. Если в результате получается число, меньшее нуля, количество баллов полагается равным нулю.

## Переменная `virtual`

<b>Имя переменной:</b>	<code>virtual</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<code>boolean</code>
<b>Может отсутствовать:</b>	<code>da</code>
<b>Значение по умолчанию:</b>	<code>false</code>
<b>Может повторяться:</b>	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет устанавливать режим виртуального турнира. Виртуальный турнир поддерживается только для типа турнира *АСМ*. Виртуальный турнир отличается тем, что отчёт времени для каждого участника ведётся независимо от других участников. Каждый участник получает возможность начать турнир в произвольный удобный для него момент времени. По истечении заданной продолжительности турнира (см. параметр `contest_time`) турнир для этого участника автоматически завершится.

**Пример.** Данный пример устанавливает режим виртуальности турнира.

```
virtual
```

## Переменная `test_sfx`

<b>Имя переменной:</b>	<b>test_sfx</b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>da</i>
<b>Значение по умолчанию:</b>	(пустая строка)
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает суффикс имён файлов, в которых хранятся входные данные тестов. Полный путь к очередному тесту определяется следующим образом:

```
path=${prob.test_dir}/${prob.short_name}/${test_num}${prob.test_sfx}
```

Здесь `prob` — это задача, решаемая тестируемой программой. `test_num` — это номер теста с тремя десятичными цифрами (включая ведущие незначащие нули). `${prob.test_dir}` — это значение конфигурационной переменной `test_dir` задачи. Если описание задачи не устанавливает эту переменную, используется глобальная переменная `test_dir`. `${prob.short_name}` — это короткое имя задачи (значение конфигурационной переменной `short_name`) описания задачи. `${prob.test_sfx}` — это значение конфигурационной переменной `test_sfx` описания задачи. Если в описании задачи эта переменная не установлена, используется глобальная переменная `test_sfx`.

### Пример.

```
test_sfx = ".dat"
```

## Переменная `corr_sfx`

Имя переменной:	<code>corr_sfx</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	(пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает суффикс имён файлов, в которых хранятся правильные ответы к тестам. Полный путь к очередному правильному ответу определяется следующим образом:

```
path=${prob.corr_dir}/${prob.short_name}/${test_num}${prob.corr_sfx}
```

Здесь `prob` — это задача, решаемая тестируемой программой. `test_num` — это номер теста с тремя десятичными цифрами (включая ведущие незначащие нули). `${prob.corr_dir}` — это значение конфигурационной переменной `corr_dir` задачи. Если описание задачи не устанавливает эту переменную, используется глобальная переменная `corr_dir`. `${prob.short_name}` — это короткое имя задачи (значение конфигурационной переменной `short_name`) описания задачи. `${prob.corr_sfx}` — это значение конфигурационной переменной `corr_sfx` описания задачи. Если в описании задачи эта переменная не установлена, используется глобальная переменная `corr_sfx`.

### Пример.

```
corr_sfx = ".res"
```

## Переменная `sound_player`

Имя переменной:	<code>sound_player</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт полный путь к программе проигрывания звуковых файлов. Если переменная установлена, программа `run` проигрывает звуковые файлы, пути к которым установлены в конфигурационных переменных `accept_sound`, `runtime_sound`, `timelimit_sound`, `wrong_sound`, `presentation_sound` и `internal_sound` в зависимости от результата тестирования. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

### Пример.

```
sound_player = /usr/bin/artsplay
```

## Переменная `accept_sound`

<b>Имя переменной:</b>	<b><code>accept_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, при успешном завершении тестирования, то есть если тестируемая программа успешно прошла все тесты, будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
accept_sound = /usr/share/ejudge/sound/accept.wav
```

## Переменная `runtime_sound`

<b>Имя переменной:</b>	<b><code>runtime_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, при возникновении ошибки времени выполнения (`runtime error`) тестируемой программы будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
runtime_sound = /usr/share/ejudge/sound/runtime.wav
```

## Переменная `timelimit_sound`

<b>Имя переменной:</b>	<b><code>timelimit_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, при превышении максимального времени, отведённого на работу тестируемой программы, (time limit exceeded) будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
timelimit_sound = /usr/share/ejudge/sound/timelimit.wav
```

## Переменная `wrong_sound`

<b>Имя переменной:</b>	<b><code>wrong_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, если на некотором тесте тестируемая программа дала неправильный ответ (wrong answer), будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
wrong_sound = /usr/share/ejudge/sound/wrong.wav
```

## Переменная `presentation_sound`

<b>Имя переменной:</b>	<b><code>presentation_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, если на некотором тесте тестируемая программа дала ответ, нарушающий формат результата, (presentation error), будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
presentation_sound = /usr/share/ejudge/sound/presentation.wav
```

## Переменная `internal_sound`

<b>Имя переменной:</b>	<b><code>internal_sound</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если установлена данная конфигурационная переменная и установлена конфигурационная переменная `sound_player`, и если при тестировании возникает внутренняя ошибка программы, проверяющей ответ тестируемой программы, будет воспроизведён звуковой файл, путь к которому указан в данной конфигурационной переменной. Звуковое оповещение о результате тестирования поддерживается только в режиме турнира *АСМ*.

**Пример.**

```
internal_sound = /usr/share/ejudge/sound/internal.wav
```



## Переменная `autoupdate_standings`

Имя переменной:	<code>autoupdate_standings</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>true</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная управляет автоматическим обновлением таблицы текущего положения участников турнира. Если данная конфигурационная переменная установлена в *true*, то таблицы текущего положения участников автоматически обновляются, когда становятся известными результаты тестирования очередной программы, а также в некоторых других случаях, за исключением времени «заморозки» (см. конфигурационные переменные `board_fog_time`, `board_unfog_time`). Кроме этого администратор турнира имеет возможность в любой момент обновить таблицу результатов. Если значение данной конфигурационной переменной установлено в *false* с помощью строки

```
autoupdate_standings = 0
```

то таблица текущего положения никогда не обновляется автоматически. Возможность обновлять таблицу текущих результатов с помощью программы-администратора турнира сохраняется.

## Переменная `standings_file_name`

Имя переменной:	<code>standings_file_name</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>имя файла</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>standings.html</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная определяет имя файла, в который выводится таблица текущего положения участников в турнире. Файл таблицы результатов таким образом создаётся с полным путём

```
${status_dir}/dir/${standings_file_name}
```

Где `${status_dir}` — значение конфигурационной переменной `status_dir`. Чтобы сделать таблицу текущих результатов доступной для участников и зрителей, нужно создать символическую ссылку из дерева каталогов `html`-сервера на этот файл. При этом `http`-сервер должен поддерживать символические ссылки в том каталоге, в котором создана ссылка на таблицу текущих результатов. Например, для сервера `Apache` это достигается заданием опции `FollowSymLinks` в его конфигурационном файле или в файле `.htaccess`.

```
standings_file_name = "standings.shtml"
```

## Переменная `stand_header_file`

<b>Имя переменной:</b>	<b><code>stand_header_file</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если данная конфигурационная переменная определена, то при генерации таблицы текущего положения участников в начало генерируемого файла будет помещаться текст, находящийся в файле-заголовке, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-заголовка считывается один раз при старте программы `serve`.

```
stand_header_file = "/home/httpd/ssi/stand_head.shtml"
```

## Переменная `stand_footer_file`

<b>Имя переменной:</b>	<b><code>stand_footer_file</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если данная конфигурационная переменная определена, то при генерации таблицы текущего положения участников в конец генерируемого файла будет помещаться текст, находящийся в файле-подвале, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-подвала считывается один раз при старте программы `serve`.

```
stand_footer_file = "/home/httpd/ssi/stand_foot.shtml"
```

## Переменная `stand2_file_name`

<b>Имя переменной:</b>	<b><code>stand2_file_name</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлена</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если значение данной конфигурационной переменной определено, программа `serve` генерирует второй файл таблицы текущего положения участников турнира. Второй файл таблицы результатов создаётся со следующим полным путём:

```
${status_dir}/dir/${stand2_file_name}
```

Где `${status_dir}` — значение конфигурационной переменной `status_dir`. Чтобы сделать таблицу текущих результатов доступной для участников и зрителей, нужно создать символическую ссылку из дерева каталогов `html`-сервера на этот файл. При этом `http`-сервер должен поддерживать символические ссылки в том каталоге, в котором создана ссылка на таблицу текущих результатов. Например, для сервера `Apache` это достигается заданием опции `FollowSymlinks` в его конфигурационном файле или в файле `.htaccess`. Необходимость двух отдельных файлов текущего положения команд обосновывается тем, что каждый из этих файлов может иметь независимое друг от друга дизайнерское оформление. Например, главный файл (см. переменную `standings_file_name`) может быть оформлен в контексте сайта олимпиады, а вспомогательный — чтобы демонстрироваться с помощью проектора.

```
stand2_file_name = "standings2.shtml"
```

## Переменная `stand2_header_file`

<b>Имя переменной:</b>	<b><code>stand2_header_file</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если данная конфигурационная переменная и переменная `stand2_file_name` определены, то при генерации вспомогательной таблицы текущего положения участников в начало генерируемого файла будет помещаться текст, находящийся в файле-заголовке, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-заголовка считывается один раз при старте программы `serve`.

```
stand2_header_file = "/home/httpd/ssi/stand2_head.shtml"
```

## Переменная `stand2_footer_file`

<b>Имя переменной:</b>	<b><code>stand2_footer_file</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если данная конфигурационная переменная и переменная `stand2_file_name` определены, то при генерации вспомогательной таблицы текущего положения участников в конец генерируемого файла будет помещаться текст, находящийся в файле-подвале, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-подвала считывается один раз при старте программы `serve`.

```
stand2_footer_file = "/home/httpd/ssi/stand2_foot.shtml"
```

## Переменная `plog_file_name`

<b>Имя переменной:</b>	<b><code>plog_file_name</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет задать имя файла, для генерируемого html-файла с журналом посылок. Журнал посылок содержит информацию о каждой посылке, которая включает в себя номер посылки, участника, задачу, язык и результат. Если значение переменной установлено, журнал посылок генерируется в файл со следующим путём:

```
${status_dir}/dir/${plog_file_name}
```

Где `${status_dir}` — значение конфигурационной переменной `status_dir`. Чтобы сделать журнал посылок доступным для участников и зрителей, нужно создать символическую ссылку из дерева каталогов html-сервера на этот файл. При этом http-сервер должен поддерживать символические ссылки в том каталоге, в котором создана ссылка на журнал посылок. Например, для сервера Apache это достигается заданием опции `FollowSymLinks` в его конфигурационном файле или в файле `.htaccess`.

```
plog_file_name = "log.shtml"
```

## Переменная `plog_update_time`

Имя переменной:	<code>plog_update_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	30
Может повторяться:	<i>нет</i>

**Описание.** Если установлена конфигурационная переменная `plog_file_name`, данная конфигурационная переменная позволяет задать период в секундах обновления журнала посылок турнира. В отличие от таблицы положения участников турнира, которая обновляется при поступлении новой информации, журнал посылок турнира обновляется всегда через один и тот же промежуток времени, задаваемый данной конфигурационной переменной.

**Пример.**

```
plog_update_time = 60
```

## Переменная `plog_header_file`

Имя переменной:	<code>plog_header_file</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>имя файла</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная и переменная `plog_file_name` определены, то при генерации журнала посылок турнира в начало генерируемого файла будет помещаться текст, находящийся в файле-заголовке, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-заголовка считывается один раз при старте программы `serve`.

```
plog_header_file = "/home/httpd/ssi/plog_head.shtml"
```

## Переменная `plog_footer_file`

<b>Имя переменной:</b>	<b><code>plog_footer_file</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>имя файла</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Если данная конфигурационная переменная и переменная `plog_file_name` определены, то при генерации журнала посылка турнира в конец генерируемого файла будет помещаться текст, находящийся в файле-подвале, полный путь к которому задаётся в этой конфигурационной переменной. Содержимое файла-подвала считывается один раз при старте программы `serve`.

```
plog_footer_file = "/home/httpd/ssi/plog_foot.shtml"
```

## Переменная `team_info_url`

<b>Имя переменной:</b>	<code>team_info_url</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>шаблон URL</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт шаблон URL, по которому находится более подробная информация об участнике. Этот шаблон раскрывается в момент генерации таблицы положения участников в турнире и формирует гиперссылку от имени участника и его идентификатора к отдельной странице с подробной информацией о команде. Подробная информация о поддерживаемых форматных преобразованиях дана в разделе [2.11.2](#).

**Пример.**

```
team_info_url = "http://www.contest.ru/users/%Mi"
```

## Переменная `prob_info_url`

<b>Имя переменной:</b>	<code>prob_info_url</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>шаблон URL</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт шаблон URL, по которому находится текст условия задачи. Этот шаблон раскрывается в момент генерации заголовка таблицы текущего положения участников в турнире и формирует гиперссылку от идентификатора задачи к отдельной странице с текстом этой задачи. Подробная информация о поддерживаемых форматных преобразованиях дана в разделе [2.11.2](#).

**Пример.**

```
prob_info_url = "http://www.contest.ru/problems/%lPs.shtml"
```

## Переменная `team_enable_src_view`

Имя переменной:	<code>team_enable_src_view</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, участники турнира получают возможность просматривать текст программ, посланных ими на проверку и хранящихся в базе данных системы `ejudge`.

**Пример.**

```
team_enable_src_view
```

## Переменная `team_enable_rep_view`

Имя переменной:	<code>team_enable_rep_view</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, участники турнира получают возможность просматривать сокращённую версию протокола тестирования для программ, посланных ими на проверку и хранящихся в базе данных системы `ejudge`. Сокращённая (пользовательская) версия протокола тестирования содержит только информацию о прохождении программой каждого теста, но не содержит самих тестов, результатов, выдаваемых программой, и правильных ответов.

**Пример.**

```
team_enable_rep_view
```



## Переменная `report_error_code`

Имя переменной:	<code>report_error_code</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>nem</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, и конфигурационная переменная `team_enable_rep_view` установлена в *true*, то сокращённая версия протокола тестирования будет содержать значение кода возврата, выработанное тестируемой программой на каждом тесте. В противном случае в сокращённом протоколе тестирования будет просто стоять отметка о том, был ли код завершения нормальным (0) или нет (любой ненулевой код). Информация о выработанном коде возврата может быть тривиальным образом использована для угадывания теста, поэтому по умолчанию эта информация отключена.

### Пример.

```
report_error_code
```

## Переменная `enable_continue`

Имя переменной:	<code>enable_continue</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>nem</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, и продолжительность турнира не ограничена (см. конфигурационную переменную `contest_time`), администратор турнира получает возможность продолжить турнир после того, так он был остановлен. Во всех других случаях остановленный или завершившийся турнир не может быть продолжен.

### Пример.

```
enable_continue
```

## Переменная `show_astr_time`

Имя переменной:	<code>show_astr_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, в таблице текущих посылок интерфейса администратора генерируемой программой `master` время посылки участником решения будет выведено как астрономическое время, а не как время от начала турнира. Данная конфигурационная переменная предназначена в первую очередь для использования в турнирах, продолжительность которых не ограничена (см. конфигурационную переменную `contest_time`).

### Пример.

```
show_astr_time
```

## Переменная `team_download_time`

Имя переменной:	<code>team_download_time</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>30</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт, как часто участники турнира могут скачивать архив своих решений. Если значение переменной установлено в 0, загрузка архива своих решений участниками отключена.

### Пример.

```
team_download_time = 600
```

## Переменная `tests_to_accept`

Имя переменной:	<code>tests_to_accept</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>1</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная определяет, на ?сколько?? тестах должна быть проверена программа участника, чтобы быть принятой на проверку. Конфигурационная переменная действует только в режиме турнира *OLYMPIAD*. Во всех других режимах турнира её значение игнорируется. Если значение этой переменной установлено в 0, для принятия задачи на проверку прохождение ею тестов не требуется. Значение данной конфигурационной переменной может быть переопределено в описании задачи с помощью конфигурационной переменной `tests_to_accept`.

### Пример.

```
tests_to_accept = 2
```

## Переменная `disable_clars`

Имя переменной:	<code>disable_clars</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, возможность участников турнира и судей и администраторов писать сообщения друг другу отключена.

### Пример.

```
disable_clars
```

## Переменная `disable_team_clars`

Имя переменной:	<code>disable_team_clars</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, участники турнира не могут писать сообщения судьям, но судьи и администраторы турнира имеют возможность отправлять сообщения всем участникам или некоторому участнику.

**Пример.**

```
disable_team_clars
```

## Переменная `ignore_compile_errors`

Имя переменной:	<code>ignore_compile_errors</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной равно `true`, если программа, посланная участником, не может быть скомпилирована (например, из-за синтаксических ошибок), такая программа не засчитывается, то есть за неё не начисляется штраф.

**Пример.**

```
ignore_compile_errors
```

## Переменная `ignore_duplicated_runs`

Имя переменной:	<code>ignore_duplicated_runs</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная имеет значение `true`, в последовательности из нескольких идущих подряд (для одного участника) посылок одной и той же программы для решения одной и той же задачи учитывается только первая посылка. Все остальные послышки игнорируются. Идентичность программ определяется с помощью сравнения их хэш-кода, полученного с помощью алгоритма SHA1.

Данная конфигурационная переменная может использоваться для удобства участников, работающих через очень медленный канал связи, так как нажатие кнопки “Reload” браузера в некоторых (редких) случаях может приводить к повторной отправке решения.

### Пример.

```
ignore_duplicated_runs
```

## Переменная `max_line_length`

Имя переменной:	<code>max_line_length</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>4096</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт ограничение на максимальную длину строки в файле протокола тестирования программы. Если длина какой-либо строки в файле, который копируется в файл протокола тестирования, превышает это значение, такая строка заменяется на строку `<string is too long>`. Ограничение максимальной длины строки действует для файлов с входными данными тестов, файлов с правильными ответами к тестам, файлов с выводом тестируемой программы, файлов с выводом в поток ошибок тестируемой программы и программы, проверяющей правильность решения.

### Пример.

```
max_line_length = 1024
```

## Переменная `max_file_length`

Имя переменной:	<code>max_file_length</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>65535</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт максимальный размер файла, копируемого в протокол тестирования программы. Если длина какого-либо файла превышает этот предел, содержимое файла заменяется на строку `<file is too long>`. Ограничение максимальной длины файла действует для файлов с входными данными тестов, файлов с правильными ответами к тестам, файлов с выводом тестируемой программы, файлов с выводом в поток ошибок тестируемой программы и программы, проверяющей правильность решения.

### Пример.

```
max_file_length = 131072
```

## Переменная `auto_short_problem_name`

Имя переменной:	<code>auto_short_problem_name</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить режим, в котором короткое имя задачи будет генерироваться по её идентификатору. Если данная конфигурационная переменная имеет значение `true`, для каждой задачи короткое имя задачи строится по идентификатору задачи как запись идентификатора в десятичной системе с пятью цифрами, включая незначащие нули. Так, для задачи с идентификатором задачи 53 будет автоматически сгенерировано короткое имя 00053.

### Пример.

```
auto_short_problem_name
```

## Переменная `checker_real_time_limit`

Имя переменной:	<code>checker_real_time_limit</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>30</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт ограничение астрономического времени на проверку результата работы тестируемой программы на очередном тесте. Другими словами, эта переменная ограничивает максимальное время работы проверяющей программы. В случае, если максимальное время работы проверяющей программы превышено, проверка программы участника завершается со статусом “Check failed” («Проверка не удалась»). Значение 0 означает отсутствие ограничения времени.

### Пример.

```
checker_real_time_limit = 60
```

## Переменная `compile_real_time_limit`

Имя переменной:	<code>compile_real_time_limit</code>
Содержится в:	<code>global</code>
Используются:	<code>compile</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>30</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт ограничение астрономического времени на компиляцию программы участника. В случае, если максимальное время компиляции превышено, проверка программы участника завершается со статусом “Check failed” («Проверка не удалась»). Значение 0 означает отсутствие ограничения времени.

### Пример.

```
compile_real_time_limit = 60
```

## Переменная `show_deadline`

Имя переменной:	<code>show_deadline</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в *true* и описание некоторой задачи содержит крайнее время сдачи (параметр `deadline`), то крайнее время сдачи будет показано в меню выбора задачи, показываемое CGI-программой `team`.

### Пример.

```
show_deadline
```



## Переменная `variant_map_file`

<b>Имя переменной:</b>	<code>variant_map_file</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>путь к файлу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к файлу, в котором находится отображение регистрационных имён участников турнира в номера вариантов для всех вариантных задач данного турнира (см. конфигурационную переменную `variant_num`). Если в турнире есть хотя бы одна вариантная задача, параметр `variant_map_file` должен быть установлен в имя корректного файла. Полный путь к файлу вариантов формируется по следующим правилам: если значение переменной `variant_map_file` начинается с символа `'/'`, то есть является абсолютным путём, используется значение этой переменной. В противном случае полный путь к файлу вариантов получается конкатенацией значения конфигурационной переменной `conf_dir` и значения переменной `variant_map_file`.

Текущая версия системы `ejudge` поддерживает следующий формат файла отображения вариантов. Файл имеет следующую структуру.

```
<variant_map version="1">
VARIANT_MAP_LINE*
</variant_map>
```

Здесь первая и последние строки должны присутствовать точно в указанном виде, `VARIANT_MAP_LINE` — строка отображения вариантов для одного пользователя. Комментарии в файле начинаются с символа `#` и оканчиваются концом строки. Пустые строки в файле игнорируются. Файл вариантов может содержать произвольное количество строк отображения вариантов, однако участник, не упомянутый в файле отображения вариантов, теряет возможность сдавать вариантные задачи. Попытка сдачи им вариантной задачи завершится ошибкой недопустимой задачи. Каждый участник может быть упомянут в файле вариантов не более одного раза.

Строка отображения вариантов одного пользователя имеет следующий вид:

```
LOGIN VARIANT*
```

Здесь `LOGIN` — это регистрационное имя пользователя, `VARIANT` — номер варианта. Номера вариантов должны быть перечислены для всех вариантных задач турнира и в том же порядке, в котором заданы вариантные задачи. Количество номеров вариантов в строке отображения вариантов должно совпадать с количеством вариантных задач турнира. Вариант задачи — это целое число от 1 и до значения конфигурационной переменной `variant_num` соответствующей задачи включительно.

### Пример.

```
variant_map_file = "variant.map"
```

Если в турнире определены три вариантные задачи, каждая из которых имеет по 4 варианта, то следующий пример задаёт отображение вариантов для пользователей `user1`, `user2`, `user3`.

```
<variant_map version="1">  
# Задачи: А, В, С  
user1      1  3  2  
user2      3  4  1  
user3      4  2  4  
</variant_map>
```

## Переменная `disable_auto_testing`

Имя переменной:	<code>disable_auto_testing</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная установлена в `true`, автоматическое тестирование решений задач, посылаемых участниками, отключено. В этом случае решение участника получает статус “Accepted for testing” и не отправляется на компиляцию и тестирование. Администратор турнира должен явно запустить тестирование решения участника, выбрав пункт “Rejudge” для данного решения.

Данная конфигурационная переменная устанавливает значение, используемое в случае, если описание задачи не определяет конфигурационную переменную `disable_auto_testing` самостоятельно.

**Пример.**

```
disable_auto_testing
```

## Переменная `enable_runlog_merge`

Имя переменной:	<code>enable_runlog_merge</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, для администратора турнира становятся доступными средства слияния журналов посылок. В этом случае появляется новый элемент управления, генерируемый программой `master`. Используя его, администратор задаёт имя файла, содержащего журнал посылок в формате XML. Этот журнал посылок далее пересылается серверу турнира, который сливает его с текущим журналом посылок. Импорт возможен, только если в текущий момент в турнире нет непроверенных посылок. При импорте корректно объединяются идентичные записи в текущем и импортируемом журналах. При этом если некоторая запись имеет флаг «локальной», её данные (результат тестирования, количество набранных баллов) имеют приоритет над данными импортируемой записи. Если импортируемая запись имеет флаг «авторитетной», то её данные имеют приоритет над локальными данными. Если некоторая запись не является ни локальной, ни авторитетной, локальные данные имеют приоритет.

В результате импорта идентификаторы посылок могут измениться. При этом идентификатор с большим номером всегда имеет время отправки не меньшее, чем идентификатор с меньшим номером. Если несколько посылок имеют одинаковое время отправки, они упорядочиваются в порядке возрастания идентификаторов пользователей. Проимпортированные записи помечаются символом \* рядом с номером посылки.

При экспорте журнала посылок локальные записи помечаются как авторитетные, что позволяет по ходу турнира обмениваться текущими журналами посылок.

## Переменная `max_cmd_length`

<b>Имя переменной:</b>	<code>max_cmd_length</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<code>integer</code>
<b>Может отсутствовать:</b>	<code>да</code>
<b>Значение по умолчанию:</b>	<code>256</code>
<b>Может повторяться:</b>	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает максимальный размер аргументов командной строки для добавления в протокол тестирования. Если размер аргументов командной строки превышает это значение, они не добавляются в протокол тестирования.

## Переменная `info_dir`

Имя переменной:	<code>info_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранится дополнительная информация к тестам для всех задач данного турнира. Сами файлы с дополнительной информацией находятся в подкаталогах этого каталога. Дополнительная информация для некоторой задачи с кратким именем `A` (задаваемом конфигурационной переменной `short_name`) находятся в подкаталоге `A` каталога дополнительной информации. Формат файлов с дополнительной информацией о тестах описывается в разделе [2.12](#). Полный путь к каталогу с дополнительной информацией определяется по следующим правилам:

- Если значение переменной `info_dir` не задано, используется значение `info`.
- Если значение переменной `info_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу скриптов, этот путь используется без изменений.
- Если значение переменной `info_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `info_dir`.

**Пример.** В следующем примере путь к каталогу тестов устанавливается в `${root_dir}/conf/../tests`, где `${root_dir}` — значение конфигурационной переменной `root_dir`. Этот путь эквивалентен пути `${root_dir}/tests`. Таким образом, тесты к задачам и дополнительная информация о тестах находятся в одном каталоге.

```
info_dir = ../tests
```

## Переменная `tgz_dir`

Имя переменной:	<code>tgz_dir</code>
Содержится в:	<code>global</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором хранятся архивы рабочих каталогов для тестирования программ в формате `.tgz` для всех задач данного турнира. Непосредственно архивы рабочих каталогов находятся в подкаталогах этого каталога. Архив рабочего каталога для некоторой задачи с кратким именем А (задаваемом конфигурационной переменной `short_name`) находятся в подкаталоге А каталога архивов. Полный путь к каталогу с дополнительной информацией определяется по следующим правилам:

- Если значение переменной `tgz_dir` не задано, используется значение `info`.
- Если значение переменной `tgz_dir` начинается с символа `'/'`, то есть значение переменной задаёт полный абсолютный путь к каталогу архивов, этот путь используется без изменений.
- Если значение переменной `tgz_dir` не начинается с символа `'/'`, полный путь к каталогу скриптов образуется конкатенацией значения переменной `conf_dir` и значения переменной `tgz_dir`.

**Пример.** В следующем примере путь к каталогу архивов устанавливается в `${root_dir}/conf/../tests`, где `${root_dir}` — значение конфигурационной переменной `root_dir`. Этот путь эквивалентен пути `${root_dir}/tests`. Таким образом, тесты к задачам и архивы рабочего каталога тестирования находятся в одном каталоге.

```
tgz_dir = ../tests
```

## Переменная `info_sfx`

<b>Имя переменной:</b>	<b><code>info_sfx</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>da</i>
<b>Значение по умолчанию:</b>	<i>.inf</i>
<b>Может повторяться:</b>	<i>nem</i>

**Описание.** Данная конфигурационная переменная устанавливает суффикс файлов с дополнительной информацией о тестах. Эти файлы находятся в подкаталогах каталога, задаваемого конфигурационной переменной `info_dir`. Формат файла с дополнительной информацией о тесте описан в разделе [2.12](#).

## Переменная `tgz_sfx`

<b>Имя переменной:</b>	<b><code>tgz_sfx</code></b>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>da</i>
<b>Значение по умолчанию:</b>	<i>.tgz</i>
<b>Может повторяться:</b>	<i>nem</i>

**Описание.** Данная конфигурационная переменная устанавливает суффикс файлов с архивами рабочего каталога тестируемых программ. Эти файлы находятся в подкаталогах каталога, задаваемого конфигурационной переменной `tgz_dir`.

## Переменная `prune_empty_users`

Имя переменной:	<code>prune_empty_users</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, участники турнира, которые не сделали ни одной попытки решения задачи, будут исключены из таблицы результатов. Если данная конфигурационная переменная установлена в *false* (значение по умолчанию), все участники турнира отображаются в таблице текущих результатов. Данная конфигурационная переменная может использоваться в турнирах с открытой регистрацией (в интернет-турнирах или виртуальных турнирах), чтобы исключить участников, которые зарегистрировались, но так и не приняли участие в турнире.



## Переменная `stand_table_attr`

Имя переменной:	<code>stand_table_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>" "</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты элемента `<table>` таблицы текущих результатов. Значение данной переменной и элемент `table` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_table_attr = " width=100%"
```

Такое значение данной конфигурационной переменной приведёт к тому, что таблица текущих результатов будет открываться тегом

```
<table width=100%>
```

**Обратите внимание,** что в текущей версии системы `ejudge`, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_place_attr`

Имя переменной:	<code>stand_place_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>" "</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбца «место» (1-й столбец) таблицы результатов турнира. Значение данной переменной добавляется к элементу `<th>` заголовка столбца таблицы и к каждому элементу `<td>`, находящемуся в столбце «место». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_place_attr = " class=st_place"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбца «место» таблицы будет открываться тегом `<th class=st_place>`, а каждая ячейка первого столбца таблицы — тегом `<td class=st_place>`.

**Обратите внимание,** что в текущей версии системы `ejudge`, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_team_attr`

Имя переменной:	<code>stand_team_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбца «команда» (2-й столбец) таблицы результатов турнира. Значение данной переменной добавляется к элементу `<th>` заголовка столбца таблицы и к каждому элементу `<td>`, находящемуся в столбце «команда». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_team_attr = " class=st_team"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбца «команда» таблицы будет открываться тегом `<th class=st_team>`, а каждая ячейка второго столбца таблицы — тегом `<td class=st_team>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_prob_attr`

Имя переменной:	<code>stand_prob_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбцов «задача» (каждой задаче турнира соответствует один столбец) таблицы результатов турнира. Значение данной переменной добавляется к элементу `<th>` заголовка столбцов таблицы и к каждому элементу `<td>`, находящемуся в столбцах «задача». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_prob_attr = " class=st_prob"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбцов «задача» таблицы будет открываться тегом `<th class=st_prob>`, а каждая ячейка столбцов задач таблицы — тегом `<td class=st_prob>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_solved_attr`

Имя переменной:	<code>stand_solved_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>nem</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбца «решённые задачи» таблицы результатов турнира. Значение данной переменной добавляется к элементу `<th>` заголовка столбцов таблицы и к каждому элементу `<td>`, находящемуся в столбцах «решённые задачи». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_solved_attr = " class=st_solved"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбцов «решённые задачи» таблицы будет открываться тегом `<th class=st_solved>`, а каждая ячейка столбцов задач таблицы — тегом `<td class=st_solved>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_score_attr`

Имя переменной:	<code>stand_score_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>nem</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбца «баллы» таблицы результатов турнира. Данный столбец присутствует только в турнирах, проводимых по системе *OLYMPIAD* или *KIROV*. Значение данной переменной добавляется к элементу `<th>` заголовка столбцов таблицы и к каждому элементу `<td>`, находящемуся в столбцах «баллы». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_score_attr = " class=st_score"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбцов «баллы» таблицы будет открываться тегом `<th class=st_score>`, а каждая ячейка столбцов задач таблицы — тегом `<td class=st_score>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (`"`) в значении конфигурационной переменной.

## Переменная `stand_penalty_attr`

Имя переменной:	<code>stand_penalty_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить дополнительные атрибуты столбца «штраф» таблицы результатов турнира. Данный столбец присутствует только в турнирах, проводимых по системе *АСМ*. Значение данной переменной добавляется к элементу `<th>` заголовка столбцов таблицы и к каждому элементу `<td>`, находящемуся в столбцах «штраф». Значение данной переменной и элементы `td` или `th` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_penalty_attr = " class=st_penalty"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовок столбцов «штраф» таблицы будет открываться тегом `<th class=st_penalty>`, а каждая ячейка столбцов задач таблицы — тегом `<td class=st_penalty>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

## Переменная `stand_time_attr`

Имя переменной:	<code>stand_time_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить атрибуты, с которыми выводится время сдачи задачи в таблице результатов турнира по системе *АСМ*. Значение данной переменной добавляется к элементу `<div>` элементов таблицы. Значение данной переменной и элементы `div` не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

### Пример.

```
stand_time_attr = " class=st_time"
```

Такое значение данной конфигурационной переменной приведёт к тому, что время сдачи задачи будет открываться тегом `<div class=st_time>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

## Переменная `stand_extra_format`

Имя переменной:	<code>stand_extra_format</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить содержимое столбца «дополнительная информация» таблицы результатов турнира. Вывод дополнительного столбца активируется, если значение данной конфигурационной переменной не пусто. Значение данной переменной определяет [форматную подстановку](#) (см. раздел [2.11.2](#)), выполняемую при генерации значений данного столбца.

**Пример. Пример.**

```
stand_extra_format = "%Mc, %Mo"
```

## Переменная `stand_extra_legend`

Имя переменной:	<code>stand_extra_legend</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить название столбца дополнительной информации таблицы результатов турнира. Столбец дополнительной информации активируется, если значение конфигурационной переменной `stand_extra_format` не пусто.

**Пример.**

```
stand_extra_legend = "Город, Страна"
```

## Переменная `stand_extra_attr`

Имя переменной:	<code>stand_extra_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет определить атрибуты, с которыми выводится столбец дополнительной информации в таблице результатов турнира. Столбец дополнительной информации активируется, если значение конфигурационной переменной `stand_extra_format` не пусто. Значение данной переменной добавляется к

элементу <th> заголовка столбцов таблицы и к каждому элементу <td>, находящемуся в столбцах «дополнительная информация». Значение данной переменной и элементы td или th не разделяются дополнительными пробелами, поэтому значение данной переменной должно начинаться с пробельного символа.

**Пример.**

```
stand_extra_attr = " class=st_extra"
```

Такое значение данной конфигурационной переменной приведёт к тому, что заголовков столбцов «дополнительная информация» таблицы будет открываться тегом <th class=st\_extra>, а каждая ячейка столбцов задач таблицы — тегом <td class=st\_extra>.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

### Переменная `stand_self_row_attr`

Имя переменной:	<code>stand_self_row_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить дополнительные атрибуты строки таблицы результатов виртуального турнира, соответствующей текущему виртуальному участнику. Значение данной переменной добавляется к элементу <tr> этой строки.

**Пример.**

```
stand_self_row_attr = " bgcolor=#ffeeff"
```

Такое значение данной конфигурационной переменной приведёт к тому, что строка таблицы результатов будет открываться тегом <tr bgcolor=#ffeeff>.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

### Переменная `stand_v_row_attr`

Имя переменной:	<code>stand_v_row_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить дополнительные атрибуты строк таблицы результатов виртуального турнира, соответствующих виртуальным

участникам виртуального турнира. Значение данной переменной добавляется к элементу `<tr>` этой строки.

#### Пример.

```
stand_v_row_attr = " bgcolor=#eeffff"
```

Такое значение данной конфигурационной переменной приведёт к тому, что строки таблицы результатов будет открываться тегом `<tr bgcolor=#eeffff>`.

**Обратите внимание**, что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

### Переменная `stand_r_row_attr`

Имя переменной:	<code>stand_r_row_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>" "</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить дополнительные атрибуты строк таблицы результатов виртуального турнира, соответствующих не виртуальным участникам виртуального турнира. Значение данной переменной добавляется к элементу `<tr>` этой строки.

#### Пример.

```
stand_r_row_attr = " bgcolor=#ffffee"
```

Такое значение данной конфигурационной переменной приведёт к тому, что строки таблицы результатов будет открываться тегом `<tr bgcolor=#ffffee>`.

**Обратите внимание**, что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.

### Переменная `stand_u_row_attr`

Имя переменной:	<code>stand_u_row_attr</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>" "</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить дополнительные атрибуты строк таблицы результатов виртуального турнира, соответствующих участникам виртуального турнира с неопределённым статусом. Значение данной переменной добавляется к элементу `<tr>` этой строки.

#### Пример.

```
stand_u_row_attr = " bgcolor=#ffffff"
```

Такое значение данной конфигурационной переменной приведёт к тому, что строки таблицы результатов будет открываться тегом `<tr bgcolor=#ffffff>`.

**Обратите внимание,** что в текущей версии системы **ejudge**, к сожалению, невозможно использовать знак «кавычки» (") в значении конфигурационной переменной.



## Переменная `use_dir_hierarchy`

Имя переменной:	<code>use_dir_hierarchy</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>true</code>
Может повторяться:	<code>нет</code>
Версия <b>ejudge</b> :	начиная с 2.1.24.

**Описание.** Если данная конфигурационная переменная установлена в значение *true*, для оптимизации доступа к архивным файлам с посылками участников турнира, судейскими протоколами тестирования и протоколами тестирования для участников они организуются в иерархическую структуру каталогов. Если же данная конфигурационная переменная установлена в *false*, каждая категория архивных файлов будет храниться в одном каталоге. Например, все протоколы тестирования будут размещены в каталоге `var/archive/reports` каталога турнира.

При иерархическом хранении файлов файлы каждой архивной категории структурируются в дерево таким образом, чтобы каталог на каждом уровне содержал не более 32 записей. Максимальное количество файлов в архиве ограничено примерно 1 миллионом, поэтому для размещения всех файлов достаточно 3 уровней подкаталогов. Например, протокол тестирования для посылки с номером 1905 будет размещён в файле `var/archive/reports/0/1/R/001905`. Данная оптимизация позволяет существенно увеличить скорость открытия файлов для файловых систем, в которых записи в каталоге хранятся в неупорядоченном виде (например, `ext2/ext3`). По умолчанию поддержка иерархической структуры архивных каталогов включена.

Старый (до версии 2.1.24) и новый форматы хранения архивных файлов совместимы снизу вверх, то есть новая версия системы **ejudge** будет корректно открывать архивы, созданные в системах предыдущих версий. Однако если поддержка иерархической структуры включена, то все последующие посылки уже будут сохраняться в иерархической системе каталогов и таким образом станут недоступными для системы **ejudge** предыдущих версий.

### Пример.

```
use_dir_hierarchy = 0
```

## Переменная `use_gzip`

Имя переменной:	<code>use_gzip</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>true</i>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.24.

**Описание.** Если данная конфигурационная переменная установлена в *true*, все архивные файлы, размер которых превышает значения, установленного в глобальной конфигурационной переменной `min_gzip_size` (по умолчанию размер равен 4096 байт), будут сжиматься алгоритмом `gzip` и сохраняться в файле с суффиксом `.gz`. Размер по умолчанию в 4 килобайта выбран, исходя из того, что 4 Кб — размер минимальной единицы выделения дискового пространства в современных версиях `ext2/ext3` на архитектуре `ia32` (этот размер совпадает с размером страницы виртуальной памяти). Сжимать файлы меньшего размера бессмысленно, так как размер занимаемой дисковой памяти от этого не уменьшится. Если значение конфигурационной переменной установлено в *false*, то все файлы независимо от размера будут храниться в архиве в несжатом виде. По умолчанию режим сжатия включён.

### Пример.

```
use_gzip = 0
```

## Переменная `min_gzip_size`

Имя переменной:	<code>min_gzip_size</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	4096
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.24.

**Описание.** Данная конфигурационная переменная задаёт максимальный размер архивного файла, который не будет сжиматься алгоритмом `gzip`, если режим сжатия включён (см. глобальную конфигурационную переменную `use_gzip`). Значение по умолчанию является наилучшим для Linux на архитектуре `ia32` и файловой системы `ext2` или `ext3`.

## Переменная `team_enable_ce_view`

Имя переменной:	<code>team_enable_ce_view</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>
Версия <b>ejudge</b> :	начиная с 2.1.24.

**Описание.** Если данная конфигурационная переменная установлена в значение *true*, а глобальная конфигурационная переменная `team_enable_rep_view` установлена в *false*, участникам предоставляется доступ к протоколу компиляции их послыки в случае, если компиляция завершилась из-за ошибки. Для всех статусов послыки, кроме ошибки компиляции, протокол тестирования участникам недоступен. Если глобальная конфигурационная переменная `team_enable_rep_view` установлена в *true*, значение данной конфигурационной переменной не используется.

Для каждой задачи режим показа ошибок компиляции может быть переопределён с помощью конфигурационной переменной `team_enable_ce_view` секции описания задачи.

## Переменная `team_show_judge_report`

Имя переменной:	<code>team_show_judge_report</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>
Версия <b>ejudge</b> :	начиная с 2.1.24.

**Описание.** Если данная конфигурационная переменная установлена в значение *true*, и глобальная конфигурационная переменная `team_enable_rep_view` тоже установлена в *true*. Вместо сокращённой версии протокола тестирования (так называемой версии участника) участникам предоставляется доступ к полной судейской версии протокола тестирования. Судейский протокол тестирования в дополнение к протоколу тестирования для участника содержит тексты тестов, результат, выведенный программой, вывод программы на стандартный поток ошибок, вывод проверяющей программы. Если конфигурационная переменная `team_enable_rep_view` установлена в *false*, значение данной конфигурационной переменной не используется.

Для каждой задачи режим показа полного протокола может быть переопределён с помощью конфигурационной переменной `team_show_judge_report` секции описания задачи.

## Переменная `team_extra_dir`

<b>Имя переменной:</b>	<code>team_extra_dir</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>путь к каталогу</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>\${var_dir}/team_extra</code>
<b>Может повторяться:</b>	<i>нет</i>
<b>Версия ejudge:</b>	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная содержит путь к каталогу, в котором размещаются файлы сохраняемого состояния клиентов участников турнира. В настоящее время сохраняется статус просмотра каждым участником сообщений, адресованных ему или всем участникам турнира.

По умолчанию значение этой переменной равно `${var_dir}/team_extra`, где `${var_dir}` — значение глобальной конфигурационной переменной `var_dir`. Если значение этой переменной не начинается с символа `'/'`, то есть представляет собой относительный путь, полный путь к каталогу вычисляется относительно каталога `${var_dir}`. Если значение переменной `team_extra_dir` начинается с символа `'/'`, то есть является абсолютным путём, используется указанный абсолютный путь.

Каталог с файлами сохраняемого состояния клиентов участников турнира имеет иерархическую структуру. В каждом каталоге может находиться не более чем 32 подкаталогов или непосредственно файлов состояния. Дерево имеет высоту 4 (то есть три каталога, затем файл состояния). Таким образом максимальный идентификатор участника равен примерно 100000. Значения конфигурационной переменной `team_extra_dir`, устанавливаемого по умолчанию должно быть всегда достаточно, таким образом необходимость в явном задании этой переменной не должна возникать.

## Переменная `disable_testing`

Имя переменной:	<code>disable_testing</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Версия <code>ejudge</code> :	начиная с 2.1.25.

**Описание.** Если данная конфигурационная переменная установлена в *true*, тестирование всех задач, решения которых могут посылать участники, отключено. При приёме программы на проверку, посылка получает статус “Accepted for testing” и не отправляется на компиляцию и тестирование. В отличие от флага запрещения автоматического тестирования `disable_auto_testing` данная конфигурационная переменная запрещает всякое тестирование. Посылки по задачам с запрещённым тестированием не могут быть явно или неявно пересужены.

Данная конфигурационная переменная предназначена для того, чтобы вести учёт задач, автоматическая проверка которых по тем или иным причинам невозможна. Администратор турнира может явно выставить для посылок по этим задачам любой статус и установить любой балл. Выставленная таким образом информация будет отображена в таблице текущих результатов турнира.

Данная конфигурационная переменная обрабатывается также и программой `run`. Для задач, тестирование которых запрещено, в процессе инициализации программа `run` не проверяет наличия тестов, правильных ответов, проверяющих программ и т. д.

Данная глобальная конфигурационная переменная устанавливает флаг запрета тестирования для всех задач одновременно. Для каждой конкретной задачи флаг запрета тестирования может быть переопределён с помощью конфигурационной переменной `disable_testing` раздела описания задачи.

### Пример.

```
disable_testing
```

## Переменная `enable_printing`

Имя переменной:	<code>enable_printing</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Версия <code>ejudge</code> :	начиная с 2.1.26.

**Описание.** Если данная конфигурационная переменная установлена в *true*, участники турнира получают возможность печатать тексты программ, посылаемых ими на проверку, на принтере. Для печати вызывается сначала программа `a2ps`, которая генерирует PostScript-файл, содержащий заглавную страницу и текст программы, затем программа `lpr`, ставя-

щая PostScript-файл в очередь печати. Для участников турнира устанавливается максимальное число страниц, которые могут быть напечатаны (см. конфигурационную переменную `team_page_quota`).

Пользователь, для которого установлен бит привилегий `PRINT_RUN` и который имеет привилегии запускать программу `judge` или `master` и просматривать текст решения участника, может выводить текст программы на принтер вне зависимости от значения данной конфигурационной переменной.

### Переменная `team_page_quota`

<b>Имя переменной:</b>	<code>team_page_quota</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<code>integer</code>
<b>Может отсутствовать:</b>	<code>да</code>
<b>Значение по умолчанию:</b>	<code>50</code>
<b>Может повторяться:</b>	<code>нет</code>
<b>Версия <code>ejudge</code>:</b>	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная устанавливает максимальное число страниц, которые могут быть напечатаны на принтере непривилегированным участником турнира. Переменная действует, только если глобальная конфигурационная переменная `enable_printing` установлена в `true`.

Количество страниц занимаемых распечаткой текста программы вычисляется по результату работы программы `a2ps` с помощью анализа диагностики, выводимой программой на стандартный поток ошибок. В количество страниц распечатки включается и обязательная заглавная страница, содержащая информацию о данной посылке. Таким образом, минимальное количество страниц, занимаемых распечаткой равно 2. Если при генерации очередного PostScript-файла обнаруживается, что общее количество напечатанных страниц с учётом данного файла превышает квоту, выдаётся сообщение об ошибке превышения квоты печати.

#### Пример.

```
team_page_quota = 20
```

### Переменная `a2ps_path`

<b>Имя переменной:</b>	<code>a2ps_path</code>
<b>Содержится в:</b>	<code>global</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>путь к программе</i>
<b>Может отсутствовать:</b>	<code>да</code>
<b>Значение по умолчанию:</b>	<code>/usr/bin/a2ps</code>
<b>Может повторяться:</b>	<code>нет</code>
<b>Версия <code>ejudge</code>:</b>	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная позволяет задать путь к программе `a2ps`, которая используется для преобразования текста программы в формат PostScript для печати на принтере.

#### Пример.

```
a2ps_path = "/usr/local/bin/a2ps"
```

## Переменная **a2ps\_args**

<b>Имя переменной:</b>	<b>a2ps_args</b>
<b>Содержится в:</b>	<a href="#">global</a>
<b>Используются:</b>	<b>serve</b>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>см. ниже</i>
<b>Может повторяться:</b>	<i>да</i>
<b>Версия ejudge:</b>	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная позволяет задать дополнительные аргументы для вызова программы **a2ps**, которая используется для преобразования текста программы в формат PostScript для печати на принтере. В конфигурационном файле может присутствовать несколько определений этой переменной. Каждое определение задаёт очередной аргумент для передачи программе **a2ps**. Если эта переменная не установлена, программа **a2ps** будет вызвана со следующими аргументами:

```
/usr/bin/a2ps -l -E -o PSFILE FILES
```

Здесь PSFILE — это путь к временному PostScript-файлу, который далее будет отправлен на печать, а FILES — временные файлы, созданные программой **serve**. Первый файл — это титульная страница, а второй файл — файл с текстом программы.

### **Пример.**

```
a2ps_args = "-2"  
a2ps_args = "-E"  
a2ps_args = "-X"  
a2ps_args = "koi8-r"
```

В данном случае программа **a2ps** будет вызвана следующим образом:

```
/usr/bin/a2ps -2 -E -X koi8-r -o PSFILE FILES
```

## Переменная **lpr\_path**

<b>Имя переменной:</b>	<b>lpr_path</b>
<b>Содержится в:</b>	<a href="#">global</a>
<b>Используются:</b>	<b>serve</b>
<b>Тип содержимого:</b>	<i>путь к программе</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>/usr/bin/lpr</i>
<b>Может повторяться:</b>	<i>нет</i>
<b>Версия ejudge:</b>	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная позволяет задать путь к программе **lpr**, которая используется для постановки PostScript-файла в очередь печати на принтер.

### **Пример.**

```
lpr_path = "/usr/local/bin/lpr"
```

## Переменная `lpr_args`

Имя переменной:	<b><code>lpr_args</code></b>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>да</i>
Версия <b>ejudge</b> :	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная позволяет устанавливать дополнительные параметры, которые будут переданы программе постановки задания в очередь печати `lpr`. Если данная переменная не установлена, программа `lpr` запускается следующим образом:

```
/usr/bin/lpr PSFILE
```

Здесь `PSFILE` — имя временного PostScript-файла с заданием печати (см. описание глобальной конфигурационной переменной `a2ps_args`). Определение переменной `lpr_args` может повторяться несколько раз. Каждое определение задаёт один аргумент командной строки для передачи программе `lpr`.

### Пример.

```
lpr_args = "-P"  
lpr_args = "myprinter"
```

В данном случае программа `lpr` будет вызываться следующим образом:

```
/usr/bin/lpr -P myprinter PSFILE
```

## Переменная `diff_path`

Имя переменной:	<b><code>diff_path</code></b>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к программе</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<code>/usr/bin/diff</code>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.27.

**Описание.** Данная конфигурационная переменная позволяет задать путь к программе `diff`, которая используется для сравнения текстов двух посылок.

### Пример.

```
lpr_path = "/usr/local/bin/diff"
```

## Переменная `enable_report_upload`



<b>Имя переменной:</b>	<b>enable_report_upload</b>
<b>Содержится в:</b>	<a href="#">global</a>
<b>Используются:</b>	<b>serve</b>
<b>Тип содержимого:</b>	<i>boolean</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>false</i>
<b>Может повторяться:</b>	<i>нет</i>
<b>Версия ejudge:</b>	начиная с 2.1.28

**Описание.** Если данная конфигурационная переменная установлена в *true*, пользователь с помощью CGI-программ **master** или **judge** получает возможность загружать на сервер турнира протокол тестирования. Для этого пользователя в соответствующем турнире должен быть установлен бит полномочий EDIT\_RUN. Если в турнире включена поддержка протоколов тестирования участников (см. глобальную конфигурационную переменную [team\\_enable\\_rep\\_view](#) при загрузке протокола на сервер можно выбрать, заместит ли загружаемый протокол протокол участника, судьи или сразу оба.

## Переменная **priority\_adjustment**

<b>Имя переменной:</b>	<b>priority_adjustment</b>
<b>Содержится в:</b>	<a href="#">global</a>
<b>Используются:</b>	<b>serve</b>
<b>Тип содержимого:</b>	<i>integer</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	0
<b>Может повторяться:</b>	<i>нет</i>
<b>Версия ejudge:</b>	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет установить базовый приоритет проверки решений. Приоритет проверки решений изменяется от -16 (самый высокий приоритет) до 15 (самый низкий приоритет). Значение, установленное в данной конфигурационной переменной применяется как базовое значение приоритета для всех посылок данного турнира. Базовое значение приоритета может модифицироваться в зависимости от идентификатора пользователя (глобальная конфигурационная переменная [user\\_priority\\_adjustment](#)), сдаваемой задачи (конфигурационная переменная [priority\\_adjustment](#) секции описания задачи), используемого языка программирования (конфигурационная переменная [priority\\_adjustment](#) секции описания языка программирования) и тестировщика для данной комбинации задачи и языка (конфигурационная переменная [priority\\_adjustment](#) секции описания тестировщика). Полное значение приоритета получается сложением глобального приоритета, приоритета задачи, приоритета языка программирования и приоритета тестировщика.

Программа **run** выбирает запросы на тестирование из каталога запросов в зависимости от приоритета запроса. Более приоритетный запрос обрабатывается первым. Если во время тестирования некоторой посылки поступает более приоритетная посылка, тестирование текущей посылки доводится до конца. Обратите внимание, что приоритет тестирования не имеет отношения к приоритету процесса (т. н. *nice*), с которым работает программа тестирования.

## Переменная `user_priority_adjustment`

Имя переменной:	<code>user_priority_adjustment</code>
Содержится в:	<code>global</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>спецификация приоритета</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>да</i>
Версия <code>ejudge</code> :	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет задать изменение приоритета тестирования (см. описание глобальной конфигурационной переменной `priority_adjustment`) для некоторого пользователя. Определение данной переменной может быть повторено несколько раз, задавая таким образом изменения приоритета для нескольких пользователей.

Спецификация приоритета имеет следующий вид:

```
LOGIN PRIO-CHANGE
```

Здесь *LOGIN* — это регистрационное имя пользователя, а *PRIO-CHANGE* — целое число, изменение приоритета. Отрицательное изменение приоритета означает повышение приоритета относительно базового значения, а положительное — понижение приоритета.

**Пример.**

```
user_priority_adjustment = "admin -10"  
user_priority_adjustment = "vasya 5"
```

## 2.11.5 Параметры задачи

Каждая задача описывается в отдельной секции с именем `problem`. Система `ejudge` в настоящее время поддерживает до 100 задач. Чтобы увеличить максимальное число одновременно поддерживаемых задач необходимо изменить константу `MAX_PROBLEM` в файле `prepare.c`.

Все конфигурационные переменные, доступные в описании задачи, перечислены ниже.

Название	Стр.	Описание
<code>abstract</code>	166	Флаг абстрактной задачи.
<code>accept_partial</code>	194	Флаг приёма задач на тестирование даже если решение не проходит обязательные тесты.
<code>checker_real_time_limit</code>	185	Ограничение времени работы проверяющей программы.
<code>corr_dir</code>	169	Каталог с правильными ответами к тестам.
<code>corr_sfx</code>	171	Суффикс файлов с правильными ответами к тестам.
<code>date_penalty</code>	194	Штраф в зависимости от времени сдачи.
<code>deadline</code>	186	Крайний срок сдачи задачи.
<code>disable_auto_testing</code>	188	Отключение режима автоматической проверки решения.
<code>disable_language</code>	195	Запрет использования указанных языков.
<code>disable_testing</code>	188	Запрет проверки решения средствами <code>ejudge</code> .
<code>full_score</code>	180	Количество баллов за полное решение задачи.
<code>hidden</code>	194	Флаг «скрытой» задачи.
<code>id</code>	164	Идентификатор задачи.
<code>info_dir</code>	191	Каталог файлов с дополнительной информацией о тестах.
<code>info_sfx</code>	193	Суффикс файлов с дополнительной информацией о тестах.
<code>input_file</code>	173	Имя входного файла для решений задачи.
<code>long_name</code>	165	«Длинное» имя задачи.
<code>output_file</code>	174	Имя выходного файла для решений задачи.
<code>priority_adjustment</code>	196	Поправка к приоритету тестирования для задачи.
<code>real_time_limit</code>	179	Лимит астрономического времени на 1 тест для решения.
<code>run_penalty</code>	181	Штраф за попытку решения задачи.
<code>short_name</code>	165	«Короткое» имя задачи.
<code>super</code>	166	Имя наследуемой абстрактной задачи.
<code>team_enable_ce_view</code>	176	Флаг разрешения просмотра участником протокола компиляции программы в случае ошибки компиляции.
<code>team_enable_rep_view</code>	175	Флаг разрешения просмотра участниками протокола тестирования.
<code>team_show_judge_report</code>	176	Флаг доступа участников к судейскому протоколу тестирования.
<code>tester_id</code>	164	Идентификатор задачи для передачи на тестирование.

*Продолжение на следующей странице*

Название	Стр.	Описание
<code>test_dir</code>	168	Каталог с тестами к задаче.
<code>test_score</code>	182	Количество баллов за один тест.
<code>test_score_list</code>	183	Спецификация баллов за тесты.
<code>test_sets</code>	184	Спецификация баллов за множество тестов.
<code>test_sfx</code>	170	Суффикс имён файлов с тестами.
<code>tests_to_accept</code>	179	Количество тестов для принятия задачи в режиме турнира <i>OLYMPIAD</i> .
<code>tgz_dir</code>	192	Каталог архивов рабочего каталога тестируемых программ.
<code>tgz_sfx</code>	193	Суффикс файлов архивов рабочего каталога тестируемых программ.
<code>time_limit</code>	178	Лимит виртуального процессорного времени на 1 тест для решения.
<code>use_corr</code>	175	Флаг использования правильных ответов к тестам.
<code>use_info</code>	189	Флаг использования файлов с дополнительной информацией о тестах.
<code>use_stdin</code>	172	Флаг использования стандартного потока ввода для входных данных.
<code>use_stdout</code>	172	Флаг использования стандартного потока вывода для ответа.
<code>use_tgz</code>	190	Флаг использования файлов архивов рабочего каталога тестируемой программы.
<code>variable_full_score</code>	180	Флаг переменного максимального балла за задачу.
<code>variant_num</code>	187	Общее количество вариантов данной задачи.

## Переменная `id`

Имя переменной:	<code>id</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает идентификатор задачи. Идентификатор задачи — это целое число в интервале от 1 до константы `MAX_PROBLEM`, которая определяется в исходном файле `prepare.c` системы **ejudge**. Идентификатор задачи присваивается только неабстрактным задачам. Абстрактные задачи не имеют идентификатора. Каждая неабстрактная задача должна иметь уникальный идентификатор. Таким образом, одновременно поддерживается до неабстрактных 100 задач. Для увеличения количества одновременно поддерживаемых задач система **ejudge** должна быть перекомпилирована с большим значением константы `MAX_PROBLEM`.

Если в описании задачи её идентификатор явно не указан, идентификатор задачи назначается автоматически. Автоматически назначаемый идентификатор на 1 больше идентификатора предыдущей неабстрактной задачи. Такая политика автоматического назначения идентификаторов задачи может привести к ошибке из-за повторного использования одного и того же идентификатора для разных задач.

## Переменная `tester_id`

Имя переменной:	<code>tester_id</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная переменная задаёт идентификатор задачи для сервера тестирования `run`, если этот идентификатор отличается от идентификатора задачи в сервере турнира `serve`. По умолчанию основной идентификатор задачи и идентификатор задачи для тестирования совпадают. Данная конфигурационная переменная может применяться, если сервер тестирования настроен на тестирование задач нескольких турниров одновременно. В этом случае идентификатор задачи в конфигурационном файле сервера тестирования и идентификатор задачи в конфигурационном файле сервера турнира могут не совпадать. Конфигурационная переменная `tester_id` позволяет в этом случае установить идентификатор задачи для тестирования в описании задачи.

Эта конфигурационная переменная не наследуется от абстрактного описания задачи, поэтому её использование в абстрактных задачах не имеет смысла.

```
tester_id = 20
```

## Переменная `short_name`

Имя переменной:	<code>short_name</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает короткое имя задачи. Для абстрактных задач короткое имя задачи должно быть установлено и должно быть уникально среди всех абстрактных задач. Для неабстрактных задач короткое имя должно быть установлено, кроме случая, когда значение глобальной конфигурационной переменной `auto_short_problem_name` установлено в `true`. В этом случае если короткое имя задачи не задано, оно будет сгенерировано автоматически из идентификатора задачи, используя форматное преобразование `%05d` функции `snprintf`. Короткое имя неабстрактной задачи не обязано быть уникальным. Оно может совпадать с другим коротким именем абстрактной или неабстрактной задачи.

Короткое имя неабстрактной задачи является, по сути, идентификатором задачи для участников турнира и зрителей. Короткое имя задачи отображается в таблице текущего положения участников, в журнале посылок, в меню сдачи программы на проверку CGI-программы `team` и т. д. Традиционно короткое имя задачи — это заглавная латинская буква A, B и т. д. для основных задач турнира и Z, Y для пробных задач турнира.

Короткое имя абстрактной задачи используется для идентификации абстрактной задачи при наследовании её свойств с помощью конфигурационной переменной `super`. Короткое имя не наследуется.

```
short_name = "A"
```

## Переменная `long_name`

Имя переменной:	<code>long_name</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает «длинное» имя задачи, то есть её содержательное название. Описание абстрактной задачи не может устанавливать длинное имя задачи, длинное имя задачи не наследуется. Если описание неабстрактной задачи не устанавливает длинное имя задачи, оно генерируется автоматически по шаблону `"Problem %d"` по идентификатору задачи.

```
long_name = "За пивОм"
```

## Переменная `abstract`

Имя переменной:	<code>abstract</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, текущее описание задачи является описанием абстрактной задачи. Абстрактная задача не может иметь идентификатора задачи (`id`) или длинного имени задачи (`long_name`), но должна иметь короткое имя (`short_name`), уникальное среди всех абстрактных задач. Абстрактная задача не может наследовать свойства другой абстрактной задачи, то есть в абстрактных задачах запрещено использование конфигурационной переменной `super`. Значения конфигурационных переменных, установленные в некоторой абстрактной задаче, наследуются конкретной задачей, указавшей данную абстрактную задачу в значении своей конфигурационной переменной `super`.

Конкретная задача наследует следующие конфигурационные переменные из описания абстрактной задачи: `accept_partial`, `checker_real_time_limit`, `corr_dir`, `corr_sfx`, `disable_auto_testing`, `disable_testing`, `full_score`, `hidden`, `info_dir`, `info_sfx`, `input_file`, `output_file`, `real_time_limit`, `run_penalty`, `team_enable_ce_view`, `team_enable_rep_view`, `team_show_judge_report`, `test_dir`, `test_score`, `test_sfx`, `test_to_accept`, `tgz_dir`, `tgz_sfx`, `time_limit`, `use_corr`, `use_info`, `use_stdin`, `use_stdout`, `use_tgz`, `variable_full_score`, `variant_num`.

При этом при наследовании конфигурационных переменных `corr_dir`, `info_dir`, `input_file`, `output_file`, `test_dir` `tgz_dir` выполняется **форматная подстановка** (см. раздел 2.11.2).

**Не наследуются** следующие конфигурационные переменные абстрактной задачи: `abstract`, `deadline`, `id`, `long_name`, `short_name`, `super`, `tester_id`, `test_score_list`, `test_sets`.

## Переменная `super`

Имя переменной:	<code>super</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена, описание задачи наследует свойства от указанной абстрактной задачи. Значение данной конфигурационной переменной должно быть коротким именем абстрактной задачи.

Конфигурационные переменные, значения которых наследуются, перечислены в описании конфигурационной переменной `abstract`.

**Пример.**

```
super = "Generic_Files"
```



## Переменная `test_dir`

Имя переменной:	<code>test_dir</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>путь к каталогу или шаблон</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором находятся тесты для данной задачи. Полный путь к каталогу с тестами для каждой неабстрактной задачи определяется по следующим правилам:

1. Если конфигурационная переменная `test_dir` задачи не определена, и эта задача наследует свойства некоторой абстрактной задачи, у которой конфигурационная переменная определена, то выполняется [форматная подстановка](#) с форматом, определяемым значением переменной `test_dir` этой абстрактной задачи, и результат помещается в переменную `test_dir` данной неабстрактной задачи.
2. Если после предыдущего шага конфигурационная переменная `test_dir` всё ещё не определена, её значение устанавливается в короткое имя данной задачи (см. переменную [short\\_name](#)).
3. Если после предыдущего шага значение конфигурационной переменной не начинается с символа `'/'`, то есть не является абсолютным путём к каталогу, значение данной конфигурационной переменной добавляется к значению глобальной конфигурационной переменной `test_dir`, и результат помещается в конфигурационную переменную `test_dir` задачи. Таким образом, глобальная конфигурационная переменная `test_dir` содержит первые компоненты пути к каталогу тестов, а конфигурационная переменная `test_dir` описания задачи — последние компоненты пути к каталогу тестов.

**Пример.** Следующий пример отключает распределение тестов по подкаталогам каталога, определяемого глобальной переменной `test_dir`.

```
test_dir = "."
```

Следующий пример для описания абстрактной задачи задаёт использование подкаталога, имя которого получается преобразованием к строчным буквам короткого имени задачи ([short\\_name](#)), в каталоге, определяемом глобальной переменной `test_dir`, для тестов каждой задачи, наследующей свойства данной абстрактной задачи.

```
test_dir = "%lPs"
```

## Переменная `corr_dir`

Имя переменной:	<code>corr_dir</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>путь к каталогу или шаблон</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором правильные ответы к тестам для данной задачи. Полный путь к каталогу с правильными ответами к тестам для каждой неабстрактной задачи определяется по следующим правилам:

1. Если конфигурационная переменная `corr_dir` задачи не определена, и эта задача наследует свойства некоторой абстрактной задачи *A*, у которой конфигурационная переменная определена, то выполняется **форматная подстановка** с форматом, определяемым значением переменной `corr_dir` абстрактной задачи *A*, и результат помещается в переменную `corr_dir` данной неабстрактной задачи.
2. Если после предыдущего шага конфигурационная переменная `corr_dir` всё ещё не определена, её значение устанавливается в короткое имя данной задачи (см. переменную `short_name`).
3. Если после предыдущего шага значение конфигурационной переменной не начинается с символа `'/'`, то есть не является абсолютным путём к каталогу, значение данной конфигурационной переменной добавляется к значению глобальной конфигурационной переменной `corr_dir`, и результат помещается в конфигурационную переменную `corr_dir` задачи. Таким образом, глобальная конфигурационная переменная `corr_dir` содержит первые компоненты пути к каталогу тестов, а конфигурационная переменная `corr_dir` описания задачи — последние компоненты пути к каталогу тестов.

**Пример.** Следующий пример отключает распределение правильных ответов к тестам по подкаталогам каталога, определяемого глобальной переменной `corr_dir`.

```
corr_dir = "."
```

Следующий пример для описания абстрактной задачи задаёт использование подкаталога, имя которого получается преобразованием к строчным буквам короткого имени задачи (`short_name`), в каталоге, определяемом глобальной переменной `corr_dir`, для правильных ответов к тестам для каждой задачи, наследующей свойства данной абстрактной задачи.

```
corr_dir = "%lPs"
```

## Переменная `test_sfx`

Имя переменной:	<code>test_sfx</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт суффикс файлов с тестами к задаче. Полный путь к очередному тесту определяется следующим образом:

```
path=${test_dir}/${test_num}${test_sfx}
```

Здесь `${test_dir}` — значение конфигурационной переменной `test_dir`, `${test_num}` — номер теста, записанный с тремя цифрами, включая незначащие нули, `${test_sfx}` — значение данной конфигурационной переменной.

Значение переменной `test_sfx` определяется по следующим правилам:

1. Если описание задачи не устанавливает переменную `test_sfx`, но наследует свойства некоторой абстрактной задачи *A*, которая устанавливает переменную `test_sfx`, то используется значение, установленное в описании абстрактной задачи *A*.
2. Если после предыдущего шага значение переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `test_sfx`.
3. Если после предыдущего шага значение переменной всё ещё не определено, значение полагается равным пустой строке .

### Пример.

```
test_sfx = ".dat"
```

## Переменная `corr_sfx`

Имя переменной:	<code>corr_sfx</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт суффикс файлов с правильными ответами к тестам к задаче. Полный путь к очередному правильному ответу к тесту определяется следующим образом:

```
path=${corr_dir}/${test_num}${corr_sfx}
```

Здесь `${corr_dir}` — значение конфигурационной переменной `corr_dir`, `${test_num}` — номер теста, записанный с тремя цифрами, включая незначащие нули, `${corr_sfx}` — значение данной конфигурационной переменной.

Правильные ответы к тестам при тестировании решения участника используются, только если конфигурационная переменная `use_corr` установлена в значение *true*. В этом случае проверяющей программе третьим аргументом командной строки передаётся полный путь к файлу с правильным ответом к тесту. Если значение конфигурационной переменной `use_corr` установлено в *false*, то путь к каталогу с ответами к тестам никак не используется, и третий аргумент в тестирующую программу не передаётся.

Значение переменной `corr_sfx` определяется по следующим правилам:

1. Если описание задачи не устанавливает переменную `corr_sfx`, но наследует свойства некоторой абстрактной задачи *A*, которая устанавливает переменную `corr_sfx`, то используется значение, установленное в описании абстрактной задачи *A*.
2. Если после предыдущего шага значение переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `corr_sfx`.
3. Если после предыдущего шага значение переменной всё ещё не определено, значение полагается равным пустой строке .

### Пример.

```
corr_sfx = ".res"
```

## Переменная `use_stdin`

Имя переменной:	<code>use_stdin</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена в *true*, решение данной задачи должно считывать входные данные со стандартного потока ввода. Если же конфигурационная переменная установлена в *false*, решение должно считывать входные данные из файла, задаваемого с помощью конфигурационной переменной `input_file`.

Если значение данной переменной в описании неабстрактной задачи не определено, то наследуется значение переменной из описания соответствующей абстрактной задачи (если таковая есть). Если и после этого значение переменной не определено, используется значение по умолчанию *false*.

### Пример.

```
use_stdin
```

## Переменная `use_stdout`

Имя переменной:	<code>use_stdout</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена в *true*, решение данной задачи должно выводить результат вычислений на стандартный поток вывода. Если же конфигурационная переменная установлена в *false*, решение должно выводить результат вычислений в файл, имя которого задаётся с помощью конфигурационной переменной `output_file`.

Если значение данной переменной в описании неабстрактной задачи не определено, то наследуется значение переменной из описания соответствующей абстрактной задачи (если таковая есть). Если и после этого значение переменной не определено, используется значение по умолчанию *false*.

### Пример.

```
use_stdout
```

## Переменная `input_file`

Имя переменной:	<code>input_file</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>input</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает имя файла, из которого программы-решения данной задачи должны считывать входные данные согласно условию задачи. Эта переменная используется только, если значение конфигурационной переменной `use_stdin` равно `false`. Значение конфигурационной переменной `input_file` устанавливается по следующим правилам:

1. Если значение данной переменной в описании неабстрактной задачи не установлено, и эта неабстрактная задача наследует свойства некоторой абстрактной задачи *A*, в описании которой данная переменная установлена, то используется значение переменной из описания абстрактной задачи *A*, при этом выполняется [форматная подстановка](#).
2. Если после предыдущего шага значение переменной всё ещё не задано, используется значение по умолчанию `input`.

**Пример.** Данная строка позволяет установить имя входного файла для абстрактной или неабстрактной задачи в `input.txt`.

```
input_file = "input.txt"
```

Следующий пример, может устанавливать имя входного файла в описании абстрактной задачи. Для всех неабстрактных задач, наследующих свойства данной абстрактной задачи, имя входного файла будет зависеть от короткого имени неабстрактной задачи. Например, для задачи с коротким именем *A* имя входного файла будет установлено в `a.in`.

```
input_file = "%lPs.in"
```

## Переменная `output_file`

Имя переменной:	<code>output_file</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>output</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает имя файла, в который программы-решения данной задачи должны записывать результат вычислений согласно условию задачи. Эта переменная используется только, если значение конфигурационной переменной `use_stdout` равно `false`. Значение конфигурационной переменной `output_file` устанавливается по следующим правилам:

1. Если значение данной переменной в описании неабстрактной задачи не установлено, и эта неабстрактная задача наследует свойства некоторой абстрактной задачи *A*, в описании которой данная переменная установлена, то используется значение переменной из описания абстрактной задачи *A*, при этом выполняется [форматная подстановка](#).
2. Если после предыдущего шага значение переменной всё ещё не задано, используется значение по умолчанию `output`.

**Пример.** Данная строка позволяет установить имя входного файла для абстрактной или неабстрактной задачи в `output.txt`.

```
output_file = "output.txt"
```

Следующий пример, может устанавливать имя входного файла в описании абстрактной задачи. Для всех неабстрактных задач, наследующих свойства данной абстрактной задачи, имя входного файла будет зависеть от короткого имени неабстрактной задачи. Например, для задачи с коротким именем *A* имя входного файла будет установлено в `a.out`.

```
output_file = "%lPs.out"
```

## Переменная `use_corr`

Имя переменной:	<code>use_corr</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной установлено в *true*, при проверке решения участника турнира используется файл с правильными ответами на тесты. Полный путь к этому файлу передаётся в качестве третьего параметра при вызове программы, сравнивающей ответ, полученный программой участника, и правильный ответ. Если значение данной переменной установлено в *false*, файлы с правильными ответами не используются. В этом случае программе проверки решения передаётся только два аргумента.

Если значение данной конфигурационной переменной в описании неабстрактной задачи не установлено, оно наследуется из абстрактной задачи (если таковая определена). Если и абстрактная задача не устанавливает значение переменной `use_corr`, то используется значение по умолчанию, равное *false*.

## Переменная `team_enable_rep_view`

Имя переменной:	<code>team_enable_rep_view</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, участники турнира получают возможность просматривать протокол тестирования своей программы. Для участников генерируется специальная версия протокола тестирования, которая не содержит тестовых данных, вывода проверяемой программы на тестовых данных, вывода проверяющей программы и т. д.

Значение данной переменной для неабстрактной задачи определяется по следующим правилам:

1. Если значение данной переменной не определено, но абстрактная задача *A*, свойства которой наследуются данной задачей, устанавливает значение данной переменной, используется значение из описания абстрактной задачи *A*.
2. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `team_enable_rep_view`, если оно определено.
3. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение по умолчанию *false*.



## Переменная `team_enable_ce_view`

Имя переменной:	<code>team_enable_ce_view</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>
Версия <code>ejudge</code> :	начиная с 2.1.24

**Описание.** Если данная конфигурационная переменная установлена в `true`, а конфигурационная переменная `team_enable_rep_view` установлена в `false`, участники турнира получают возможность просматривать протокол компиляции своей программы в случае, если соответствующая посылка имеет статус «Ошибка компиляции». Во всех остальных случаях протокол тестирования участнику будет недоступен.

Значение данной переменной для неабстрактной задачи определяется по следующим правилам:

1. Если значение данной переменной не определено, но абстрактная задача *A*, свойства которой наследуются данной задачей, устанавливает значение данной переменной, используется значение из описания абстрактной задачи *A*.
2. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `team_enable_ce_view`, если оно определено.
3. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение по умолчанию `false`.

## Переменная `team_show_judge_report`

Имя переменной:	<code>team_show_judge_report</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>
Версия <code>ejudge</code> :	начиная с 2.1.24

**Описание.** Если данная конфигурационная переменная установлена в `true` и конфигурационная переменная `team_enable_rep_view` также установлена в `true`, участники турнира получают доступ к полной версии протокола тестирования, обычно доступной только привилегированным пользователям.

Значение данной переменной для неабстрактной задачи определяется по следующим правилам:

1. Если значение данной переменной не определено, но абстрактная задача  $A$ , свойства которой наследуются данной задачей, устанавливает значение данной переменной, используется значение из описания абстрактной задачи  $A$ .
2. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `team_show_judge_report`, если оно определено.
3. Если после предыдущего шага значение данной переменной всё ещё не определено, используется значение по умолчанию *false*.

## Переменная `time_limit`

Имя переменной:	<code>time_limit</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<code>0</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт процессорное виртуальное время в секундах, отведённое на работу проверяемой программы с одним тестом. Если по истечении этого времени программа не завершила работу, она убивается, и для данного теста проверяемой программы диагностируется ошибка “Time-limit exceeded”. Виртуальное процессорное время — это время, в течении которого процессор выполнял (в режиме пользователя или в режиме ядра) данную задачу. Время, затраченное в ожидании событий ввода/вывода в данном параметре не учитывается. Значение 0 данной конфигурационной переменной означает, что время выполнения тестируемой программы не ограничивается.

Данный параметр соответствует ограничению времени работы программы на одном тесте, которое указывается в условии задачи. Однако, если тестируемая программа не занимает процессор а находится в ожидании события ввода-вывода, например, выполняя чтение из стандартного потока ввода вместо чтения из файла или выполняя системный вызов `pause`, она не тратит виртуальное процессорное время. Поэтому реальное астрономическое время выполнения тестируемой программы может быть сколь угодно больше виртуального процессорного времени. Для ограничения реального астрономического времени используется конфигурационная переменная `real_time_limit`.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию 0 (отсутствие ограничений на максимальное время выполнения).

### Пример.

```
time_limit = 5
```

## Переменная `real_time_limit`

Имя переменной:	<code>real_time_limit</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	0
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт ограничение на астрономическое время выполнения тестируемой программы на одном тесте. Ограничение астрономического времени выполнения программы должно быть больше ограничения виртуального процессорного времени (см. переменную `time_limit`). Оно должно учитывать время выполнения операций ввода-вывода и загруженность тестирующей машины.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию 0 (отсутствие ограничений на максимальное время выполнения).

### Пример.

```
real_time_limit = 30
```

## Переменная `tests_to_accept`

Имя переменной:	<code>tests_to_accept</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	1
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная определяет, сколько тестов должны быть успешно пройдены программой для того, чтобы она была принята на проверку. Конфигурационная переменная действует только в режиме турнира *OLYMPIAD*. Во всех других режимах турнира её значение игнорируется. Если значение этой переменной установлено в 0, для принятия задачи на проверку прохождение ею тестов не требуется.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если после этого значение данной переменной не определено, используется значение глобальной конфигурационной переменной `tests_to_accept`. Значение глобальной конфигурационной переменной устанавливается в 1 по умолчанию, поэтому если ни глобальная переменная, ни переменная описания задачи не определены, используется значение 1.

### Пример.

```
tests_to_accept = 2
```

## Переменная `full_score`

Имя переменной:	<b>full_score</b>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	25
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает количество баллов, начисляемых за полное решение задачи, то есть за решение, которое успешно прошло все тесты. Переменная используется только в режимах турнира *KIROV* и *OLYMPIAD*. Полное описание правил начисления баллов за решение задачи см. в описании конфигурационной переменной `score_system`.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию 25.

### Пример.

```
full_score = 50
```

## Переменная `variable_full_score`

Имя переменной:	<b>variable_full_score</b>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.25.

**Описание.** Если данная конфигурационная переменная описания задачи установлена в *true*, изменяется правило вычисления баллов за полностью решённую задачу в режимах турнира *OLYMPIAD* или *KIROV*. В обычном режиме за полностью решённую задачу (то есть за посылку, которая имеет статус ОК) даётся количество баллов, определённое в конфигурационной переменной `full_score` описания задачи (далее из этого числа вычитаются штрафные баллы и т. д.). В режиме, включаемом данной конфигурационной переменной, за удачную посылку (то есть посылку, имеющую статус ОК) даётся столько баллов, сколько указано в поле `score` этой посылки. Данная конфигурационная переменная удобна в случаях, если статус ОК выставляется вручную (например, по результатам ручного тестирования), и необходимо ранжировать принятые решения по баллам.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию *false*.

## Переменная `run_penalty`

Имя переменной:	<code>run_penalty</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<code>1</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает штраф за каждую попытку решения задачи. Например, количество баллов, полученное за пятую попытку сдачи некоторой задачи, будет на  $4 * \text{run\_penalty}$  баллов меньше, чем если то же самое решение было послано с первого раза. Переменная используется только в режимах турнира *KIROV* и *OLYMPIAD*. Полное описание правил начисления баллов за решение задачи см. в описании конфигурационной переменной `score_system`.

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию `1`.

### Пример.

```
run_penalty = 2
```

## Переменная `test_score`

Имя переменной:	<code>test_score</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<code>1</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает, сколько баллов получает решение участника за успешное прохождение одного теста. Баллы, полученные за успешно пройденные тесты, суммируются. Количество баллов для каждого теста может быть определено индивидуально с помощью конфигурационной переменной `test_score_list`. Количество баллов, назначаемых за пройденное множество тестов, может быть определено с помощью конфигурационной переменной `test_sets`. Суммарное количество баллов за все тесты задачи не может превосходить значения конфигурационной переменной `full_score`.

Данная конфигурационная переменная используется только в режиме турнира *KIROV* или *OLYMPIAD* (см. конфигурационную переменную `score_system`).

Если данная конфигурационная переменная неабстрактной задачи не определена, её значение наследуется от абстрактной задачи, если абстрактная задача указана. Если и после этого значение переменной не определено, устанавливается значение по умолчанию 1.

### Пример.

```
test_score = 2
```

## Переменная `test_score_list`

Имя переменной:	<code>test_score_list</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>нет</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет устанавливать число баллов, назначаемое за успешное прохождение тестов, индивидуально для каждого теста. Данная конфигурационная переменная используется только в режиме турнира *KIROV* или *OLYMPIAD* (см. конфигурационную переменную `score_system`). Если число баллов, назначаемое за успешное прохождение некоторого теста, в конфигурационной переменной `test_score_list` не определяется, для этого теста используется значение конфигурационной переменной `test_score`. Количество баллов, назначаемых за пройденное множество тестов, может быть определено с помощью конфигурационной переменной `test_sets`.

Значение данной конфигурационной переменной `test_score_list` — это спецификация баллов за тесты, которая имеет следующий формат (грамматика записана в нотации EBNF).

```
test_score_list_spec = { test_score_spec } ;
test_score_spec = [ "[" test_number "]" ] test_score ;
test_number = NUMBER ;
test_score = NUMBER ;
```

Другими словами, спецификация баллов за тесты — это список спецификаций баллов за отдельные тесты. В спецификации баллов за тест в квадратных скобках может указываться номер теста. Если номер теста не указан, используется номер предыдущего теста, увеличенный на 1. Если номер теста не указан в первой спецификации баллов, спецификация относится к первому тесту.

Значение данной конфигурационной переменной **не наследуется**.

**Пример.** Первый пример устанавливает количество баллов за тесты с первого по пятый в 1, 2, 3, 4, 5 баллов соответственно. Количество баллов за оставшиеся тесты (если они имеются) задаётся с помощью конфигурационной переменной `test_score`.

```
test_score_list = "1 2 3 4 5"
```

Второй пример устанавливает следующее количество баллов: за первый тест — 3 балла, за 10 тест — 4 балла, за 11 тест — 5 баллов, за 15 тест — 6 баллов.

```
test_score_list = "3 [10] 4 5 [15] 6"
```



## Переменная `test_sets`

Имя переменной:	<code>test_sets</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>нет</code>
Значение по умолчанию:	<code>"</code> (пустая строка)
Может повторяться:	<code>da</code>

**Описание.** Данная конфигурационная переменная позволяет задавать количество баллов, которое получает решение задачи, давшее правильный ответ на заданном множестве тестов. Данная конфигурационная переменная используется только в режиме турнира *KIROV* или *OLYMPIAD* (см. конфигурационную переменную `score_system`).

Данная конфигурационная переменная призвана бороться с угадыванием ответов. Например, если по условию задачи требуется дать ответ в форме "YES" или "NO", решение участника может всегда давать ответ "YES" или всегда давать ответ "NO", и такое «решение» может получить частичный балл (иногда довольно большой). Тогда чтобы бороться с такими программами, множество ответов "YES" и множество ответов "NO" могут быть указаны, как множества тестов, оцениваемые в 0 баллов. Теперь чтобы получить ненулевой балл решение участника должно хотя бы один раз правильно ответить "YES" и хотя бы один раз правильно ответить "NO".

Каждое значение конфигурационной переменной `test_sets` задаёт одно множество тестов, но переменная `test_sets` может быть повторена в описании задачи несколько раз, задавая несколько множеств тестов. Значение данной конфигурационной переменной **не наследуется**.

Значением одной конфигурационной переменной `test_sets` является строка-спецификация множества тестов, имеющая следующий формат (грамматика записана в нотации EBNF).

```
test_set_spec = test_num_list "=" set_score ;
test_num_list = { test_score } ;
set_score = NUMBER ;
test_score = NUMBER ;
```

Другими словами, спецификация множества тестов состоит из перечисления номеров тестов, за которым после знака "-" следует число баллов, назначаемое за данное множество тестов. Такое правило срабатывает только если множество тестов, на которых решение участника дало правильный ответ, в точности совпадает с указанным множеством тестов.

**Пример.** В следующем примере за успешное прохождение только тестов 2, 4, 5 решение участника получает 1 балл, а за успешное прохождение тестов 1, 4, 6 — 2 балла.

```
test_sets = "2 4 5 = 1"
test_sets = "1 4 6 = 2"
```

## Переменная `checker_real_time_limit`

Имя переменной:	<code>checker_real_time_limit</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	30
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт ограничение на астрономическое время проверки результата работы тестируемой программы. Другими словами, эта переменная ограничивает максимальное время работы проверяющей программы. В случае, если максимальное время работы проверяющей программы превышено, проверка программы участника завершается со статусом “Check failed” («Проверка не удалась»). Значение 0 означает отсутствие ограничения времени.

Если значение данной конфигурационной переменной в описании неабстрактной задачи не определено, но данная неабстрактная задача наследует свойства некоторой абстрактной задачи *A*, которая определяет данную переменную, используется значение, определённое в описании абстрактной задачи *A*. Если после этого значение данной конфигурационной переменной всё ещё не определено, используется значение глобальной конфигурационной переменной `checker_real_time_limit`. Значение последней по умолчанию равно 30, следовательно и значение конфигурационной переменной уровня задачи также равно 30.

### Пример.

```
checker_real_time_limit = 60
```

## Переменная `deadline`

Имя переменной:	<code>deadline</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>календарное время</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>нет</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет установить крайнее время, после наступления которого прекращается приём указанной задачи. Данная конфигурационная переменная позволяет избирательно прекращать приём определённых задач в определённое время. Её использование имеет смысл только для турниров, продолжительность которых не ограничена. Календарное время задаётся в одном из следующих форматов:

*Y/M/D H:I:S,*

*Y/M/D H:I,*

*Y/M/D H,*

*Y/M/D,*

где *Y* — год в диапазоне от 1970 до 2038, *M* — номер месяца в диапазоне от 1 до 12, *D* — номер дня в месяце в диапазоне от 1 до 31, *H* — час в диапазоне от 0 до 23, *I* — минуты в диапазоне от 0 до 59, *S* — секунды в диапазоне от 0 до 60. Значение отсутствующих компонент времени полагается равным 0.

### **Пример.**

```
deadline = "2003/10/01 00:00:00"
```

Следующая запись полностью эквивалентна предыдущей:

```
deadline = "2003/10/01"
```

## Переменная `variant_num`

Имя переменной:	<code>variant_num</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает количество вариантов задачи. Если значение данной переменной не установлено или явно установлено в 0, текущая задача не является вариантной, то есть предусматривается только один набор тестов, правильных ответов и одна проверяющая программа. Невариантными являются, как правило, все задачи в турнирах.

В некоторых случаях (например, при проведении самостоятельных работ с автоматической проверкой решений) желательно, чтобы у разных учащихся были бы разные задачи. Для этого можно использовать вариантные задачи. Вариантная задача — это семейство задач, у которых совпадают все параметры, кроме каталога с тестами, каталога с правильными ответами и проверяющей программы. Вариант задачи — это целое число, начиная от 1. Переменная `variant_num` задаёт количество вариантов задачи, которые, таким образом, нумеруются 1, 2 и до `variant_num`.

Все параметры, необходимые для тестирования вариантной задачи, получаются из невариантных параметров следующим образом. Каталог с тестами для варианта  $n$  должен находиться по пути `${test_dir}-n`, где `${test_dir}` — значение конфигурационной переменной `test_dir` данной задачи. Другими словами, к пути к каталогу с тестами, который был бы у данной задачи, если бы она не была вариантной, приписывается номер варианта, отделённый знаком «минус». Каталог с правильными ответами к тестам для варианта  $n$  должен находиться по пути `${corr_dir}-n`, где `${corr_dir}` — значение конфигурационной переменной `corr_dir` данной задачи. Проверяющая программа должна располагаться по пути `${check_cmd}-n`, где `${check_cmd}` — значение конфигурационной переменной `check_cmd` данной задачи.

Если значение переменной `variant_num` в описании задачи не установлена, но данная задача наследует свойства абстрактной задачи, в которой значение переменной установлено, будет использовано значение переменной, установленное в описании абстрактной задачи.

### Пример.

```
variant_num = 4
```

## Переменная `disable_auto_testing`

Имя переменной:	<code>disable_auto_testing</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Может повторяться:	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной задачи установлено в `true`, автоматическое тестирование решений данной задачи отключено. Принимаемые программой `serve` решения добавляются в базу решений участников в состоянии “Accepted for Testing” и не передаются на компиляцию и тестирование. Администратор турнира может запустить компиляцию и тестирование решения, выбрав пункт меню “Rejudge”.

Если значение данной конфигурационной переменной не установлено, но описание данной задачи наследует свойства некоторой абстрактной задачи, в которой значение данной переменной установлено, используется значение из описания абстрактной задачи. Если после этого значение данной переменной всё равно не определено, используется значение глобальной конфигурационной переменной `disable_auto_testing`.

**Пример.**

```
disable_auto_testing
```

## Переменная `disable_testing`

Имя переменной:	<code>disable_testing</code>
Содержится в:	<code>problem</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Может повторяться:	<code>нет</code>
Версия <code>ejudge</code> :	начиная с 2.1.25.

**Описание.** Данная конфигурационная переменная раздела описания задачи позволяет переопределить значение глобальной конфигурационной переменной `disable_testing` для каждой конкретной задачи. Если значение данной конфигурационной переменной задачи установлено в `true`, любое тестирование задачи средствами системы `ejudge` отключено. Администратор турнира может выставить любой необходимый статус посылки по такой задаче с помощью редактирования базы посылок.

Если значение данной конфигурационной переменной не установлено, но описание данной задачи наследует свойства некоторой абстрактной задачи, в которой значение данной переменной установлено, используется значение из описания абстрактной задачи. Если после этого значение данной переменной всё равно не определено, используется значение глобальной конфигурационной переменной `disable_testing`.

## Переменная `use_info`

<b>Имя переменной:</b>	<code>use_info</code>
<b>Содержится в:</b>	<code>problem</code>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<code>boolean</code>
<b>Может отсутствовать:</b>	<code>da</code>
<b>Наследуется:</b>	<code>da</code>
<b>Значение по умолчанию:</b>	<code>false</code>
<b>Может повторяться:</b>	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной установлено в `true`, для запуска решений участников на проверку и для проверки решений используются файлы с дополнительной информацией о тесте. В противном случае никаких дополнительных файлов не используется. Файлы с дополнительной информацией о тестах содержат аргументы командной строки для запуска тестируемой программы, комментарии к тестам и т. д. Такие файлы находятся в каталоге, определяемом конфигурационной переменной `info_dir`, имеют стандартное базовое имя, получаемое форматным преобразованием `%03d` функций семейства `printf` из номера теста, и суффикс, определяемый конфигурационной переменной `info_sfx` (по умолчанию `.inf`). Формат файлов с дополнительной информацией описан в разделе 2.12.

В случае использования `.inf`-файлов проверяющая программа получает дополнительный аргумент командной строки, содержащий путь к `.inf`-файлу для данного теста.

Если значение данной конфигурационной переменной в описании неабстрактной задачи не установлено, оно наследуется из абстрактной задачи (если таковая определена).

## Переменная `use_tgz`

Имя переменной:	<code>use_tgz</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной установлено в `true`, для запуска решений участников на проверку создаётся рабочий каталог, и тестируемая программа запускается в этом файле. Исходное состояние рабочего каталога получается разархивированием архивного файла. Такие архивные файлы находятся в каталоге, определяемом конфигурационной переменной `tgz_dir`, имеют стандартное базовое имя, получаемое форматным преобразованием `%03d` функций семейства `printf` из номера теста, и суффикс, определяемый конфигурационной переменной `tgz_sfx` (по умолчанию `.tgz`).

Для теста с номером `i` `.tgz`-архив должен содержать единственный каталог с именем полученным форматным преобразованием `%03d` из номера теста `i`. Кроме этого, в каталоге `tgz_dir` должно находиться разархивированное содержимое этого архива, то есть каталог с именем `<%03d,i>`. При запуске тестируемой программы соответствующий `.tgz` архив разворачивается в каталоге тестирования (`check_dir`) и текущий каталог запускаемой программы устанавливается на этот подкаталог. Файл с кодом программы, файлы входных и выходных данных таким образом оказываются на один уровень выше текущего каталога.

В случае использования `.tgz`-файлов проверяющая программа получает дополнительный аргумент командной строки, содержащий путь к развёрнутому содержимому соответствующего `.tgz` файла, то есть к каталогу с именем `<%03d,i>` в каталоге, определяемом конфигурационной переменной `tgz_dir`.

Если значение данной конфигурационной переменной в описании неабстрактной задачи не установлено, оно наследуется из абстрактной задачи (если таковая определена).

## Переменная `info_dir`

Имя переменной:	<code>info_dir</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу или шаблон</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором находится дополнительная информация о тестах к данной задаче. Файлы с дополнительной информацией о тестах далее кратко называются `.inf`-файлы. Полный путь к каталогу с `.inf`-файлами для каждой неабстрактной задачи определяется по следующим правилам:

1. Если конфигурационная переменная `info_dir` задачи не определена, и эта задача наследует свойства некоторой абстрактной задачи *A*, у которой эта конфигурационная переменная определена, то выполняется [форматная подстановка](#) с форматом, определяемым значением переменной `info_dir` абстрактной задачи *A*, и результат помещается в переменную `info_dir` данной неабстрактной задачи.
2. Если после предыдущего шага конфигурационная переменная `info_dir` всё ещё не определена, её значение устанавливается в короткое имя данной задачи (см. переменную [short\\_name](#)).
3. Если после предыдущего шага значение конфигурационной переменной не начинается с символа `'/'`, то есть не является абсолютным путём к каталогу, значение данной конфигурационной переменной добавляется к значению глобальной конфигурационной переменной `info_dir`, и результат помещается в конфигурационную переменную `info_dir` задачи. Таким образом, глобальная конфигурационная переменная `info_dir` содержит первые компоненты пути к каталогу `.inf`, а конфигурационная переменная `info_dir` описания задачи — последние компоненты пути к каталогу `.inf`-файлов.

**Пример.** Следующий пример отключает распределение `.inf`-файлов по подкаталогам каталога, определяемого глобальной переменной `info_dir`.

```
info_dir = "."
```

Следующий пример для описания абстрактной задачи задаёт использование подкаталога, имя которого получается преобразованием к строчным буквам короткого имени задачи ([short\\_name](#)), в каталоге, определяемом глобальной переменной `info_dir`, для `.inf`-файлов для каждой задачи, наследующей свойства данной абстрактной задачи.

```
info_dir = "%lPs"
```



## Переменная `tgz_dir`

Имя переменной:	<code>tgz_dir</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу или шаблон</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к каталогу, в котором находится архив рабочего каталога для тестирования решения участника. Архивы рабочих каталогов далее кратко называются `.tgz`-файлы. Полный путь к каталогу с `.tgz`-файлами для каждой неабстрактной задачи определяется по следующим правилам:

1. Если конфигурационная переменная `tgz_dir` задачи не определена, и эта задача наследует свойства некоторой абстрактной задачи *A*, у которой эта конфигурационная переменная определена, то выполняется [форматная подстановка](#) с форматом, определяемым значением переменной `tgz_dir` абстрактной задачи *A*, и результат помещается в переменную `tgz_dir` данной неабстрактной задачи.
2. Если после предыдущего шага конфигурационная переменная `tgz_dir` всё ещё не определена, её значение устанавливается в короткое имя данной задачи (см. переменную [short\\_name](#)).
3. Если после предыдущего шага значение конфигурационной переменной не начинается с символа `'/'`, то есть не является абсолютным путём к каталогу, значение данной конфигурационной переменной добавляется к значению глобальной конфигурационной переменной `tgz_dir`, и результат помещается в конфигурационную переменную `tgz_dir` задачи. Таким образом, глобальная конфигурационная переменная `tgz_dir` содержит первые компоненты пути к каталогу `.tgz`-файлов, а конфигурационная переменная `tgz_dir` описания задачи — последние компоненты пути к каталогу `.tgz`-файлов.

**Пример.** Следующий пример отключает распределение `.tgz`-файлов по подкаталогам каталога, определяемого глобальной переменной `tgz_dir`.

```
tgz_dir = "."
```

Следующий пример для описания абстрактной задачи задаёт использование подкаталога, имя которого получается преобразованием к строчным буквам короткого имени задачи ([short\\_name](#)), в каталоге, определяемом глобальной переменной `tgz_dir`, для `.tgz`-файлов для каждой задачи, наследующей свойства данной абстрактной задачи.

```
tgz_dir = "%lPs"
```

## Переменная `info_sfx`

Имя переменной:	<code>info_sfx</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает суффикс файлов с дополнительной информацией о тестах. Если неабстрактная задача не устанавливает непосредственно данную переменную, но наследует абстрактную задачу, устанавливающую эту конфигурационную переменную, используется значение из описания абстрактной задачи. Если после этого значение конфигурационной переменной `info_sfx` всё ещё не определено, используется значение глобальной конфигурационной переменной `info_sfx`, которая по умолчанию равна `.inf`. Таким образом можно считать, что файлы с дополнительной информацией о тесте имеют по умолчанию суффикс `.inf`.

## Переменная `tgz_sfx`

Имя переменной:	<code>tgz_sfx</code>
Содержится в:	<code>problem</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает суффикс файлов архивов рабочего каталога для тестирования программ. Если неабстрактная задача не устанавливает непосредственно данную переменную, но наследует абстрактную задачу, устанавливающую эту конфигурационную переменную, используется значение из описания абстрактной задачи. Если после этого значение конфигурационной переменной `tgz_sfx` всё ещё не определено, используется значение глобальной конфигурационной переменной `tgz_sfx`, которая по умолчанию равна `.tgz`. Таким образом можно считать, что файлы с архивами рабочего каталога имеют по умолчанию суффикс `.tgz`.

## Переменная `hidden`

Имя переменной:	<b>hidden</b>
Содержится в:	<a href="#">problem</a>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.26.

**Описание.** Если данная конфигурационная переменная описания некоторой задачи установлена в *true*, эта задача помечается как «скрытая». Скрытые задачи не отображаются в таблице текущих результатов турнира, однако доступны участникам турнира для сдачи решений.

Если в описании неабстрактной задачи данная конфигурационная переменная не была определена, используется значение переменной из абстрактной задачи, наследуемой данной задачей. Если после этого значение данной конфигурационной переменной не определено, используется значение по умолчанию *false*.

## Переменная `accept_partial`

Имя переменной:	<b>accept_partial</b>
Содержится в:	<a href="#">problem</a>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.26.

**Описание.** Данная конфигурационная переменная действует только в турнирах, проводимых по системе *OLYMPIAD*. Когда турнир находится в режиме приёма решений на проверку, каждое решение, присылаемое участниками турнира, компилируется и запускается на нескольких первых тестах из набора тестов (количество таких тестов задаётся с помощью конфигурационной переменной [tests\\_to\\_accept](#)). Если данная конфигурационная переменная `accept_partial` равна *false* (значение по умолчанию), чтобы решение было принято для последующей проверки необходимо, чтобы оно прошло все эти предварительные тесты. Если данная переменная установлена в *true*, решение принимается на проверку в любом случае кроме ошибки компиляции.

Если в описании неабстрактной задачи данная конфигурационная переменная не была определена, используется значение переменной из абстрактной задачи, наследуемой данной задачей. Если после этого значение данной конфигурационной переменной не определено, используется значение по умолчанию *false*.

## Переменная `date_penalty`

<b>Имя переменной:</b>	<b>date_penalty</b>
<b>Содержится в:</b>	<i>problem</i>
<b>Используются:</b>	<i>serve</i>
<b>Тип содержимого:</b>	<i>спецификации штрафа</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Наследуется:</b>	<i>нет</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>да</i>
<b>Версия ejudge:</b>	начиная с 2.1.28

**Описание.** Данная конфигурационная переменная позволяет назначать штраф за посылку в зависимости от времени сдачи задачи. Например, посылка, принятая раньше определённой даты, не штрафуются, а после этой даты накладывается некоторый штраф и т. д. (см. пример ниже). В рамках описания одной задачи определение переменной `date_penalty` может повторяться несколько раз. Каждое новое Каждое определение добавляет новую спецификацию штрафования к уже существующим. Значение переменной `date_penalty` не наследуется от абстрактных задач. Спецификации `date_penalty` одной задачи не зависят от `date_penalty` других задач. Спецификации упорядочены и проверяются в порядке их задания в конфигурационном файле.

Каждая спецификация штрафа имеет вид

```
DATE ADD-VALUE
```

где *DATE* — календарная дата, задаваемая в стандартном для **ejudge** формате `YYYY/MM/DD [hh[:mm[:ss]]]`, *ADD-VALUE* — значение, прибавляемое к количеству баллов, полученному в результате тестирования. Получившееся значение не может быть меньше 0 (в этом случае оно устанавливается в 0) и больше максимального значения баллов за данную задачу (в этом случае оно устанавливается в максимальное количество баллов). Штрафование в зависимости от времени работает только в режиме турнира KIROV.

#### Пример.

```
date_penalty = "2004/04/05 0"
date_penalty = "2004/04/12 -1"
date_penalty = "2005/01/01 -2"
```

В данном примере если попытка по задаче послана до 5 апреля 2004 г., она не получает штрафных очков, если она послана 5-11 апреля, она получает одно штрафное очко, а если 12 апреля и позднее - 2 штрафных очка.

### Переменная `disable_language`

<b>Имя переменной:</b>	<b>disable_language</b>
<b>Содержится в:</b>	<i>problem</i>
<b>Используются:</b>	<i>serve</i>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Наследуется:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>да</i>
<b>Версия ejudge:</b>	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет указать языки программирования, которые не могут использоваться для посылок по данной задаче. Каждый раз указывается только одно короткое имя (*short\_name*) языка программирования, но переменная `disable_language` может быть использована несколько раз в описании одной задачи.

**Пример.**

```
disable_language = "gcc"  
disable_language = "g++"
```

Запрещает для некоторой задачи использование языков с коротким именем (*short\_name*) `gcc` и `g++`.

## Переменная `priority_adjustment`

<b>Имя переменной:</b>	<code>priority_adjustment</code>
<b>Содержится в:</b>	<code>problem</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>integer</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Наследуется:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	0
<b>Может повторяться:</b>	<i>нет</i>
<b>Версия <code>ejudge</code>:</b>	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет задать поправку к приоритету тестирования для данной задачи. Значение поправки будет прибавлено к базовому значению приоритета тестирования (см. описание глобальной конфигурационной переменной `priority_adjustment`) при вычислении приоритета тестирования посылки. Положительное значение поправки уменьшает приоритет, а отрицательное — увеличивает.

**Пример.**

```
priority_adjustment = 2
```

## 2.11.6 Параметры языка

Каждый поддерживаемый язык программирования описывается в секции с именем `language`. Система **ejudge** в настоящее время поддерживает до 31 языка программирования. Чтобы увеличить количество одновременно поддерживаемых языков программирования необходимо изменить константу `MAX_LANGUAGE` в файле `prepare.c` в исходных текстах системы и перекомпилировать систему.

Название секции `language` можно считать не очень удачным, так как каждая секция описывает не язык программирования, а компилятор языка программирования. Например, конфигурационный файл может содержать 4 секции, соответствующие языку Паскаль: для **GNU Pascal**, **Free Pascal**, **Delphi**, **Borland Pascal**.

Для секций описания языка наследование свойств не поддерживается.

Все конфигурационные переменные, допустимые в секции описания языка программирования, приведены ниже.

Название	Стр.	Описание
<code>arch</code>	201	Архитектура компилятора.
<code>binary</code>	203	Флаг бинарного режима языка программирования.
<code>cmd</code>	202	Команда запуска компилятора.
<code>compile_dir</code>	203	Каталог обмена программ <b>serve</b> и <b>compile</b> .
<code>compile_id</code>	198	Идентификатор языка для передачи программе <b>compile</b> .
<code>compile_real_time_limit</code>	203	Ограничение времени на работу компилятора.
<code>disabled</code>	199	Флаг запрета использования данного компилятора.
<code>exe_sfx</code>	202	Суффикс исполняемых файлов, генерируемых компилятором.
<code>id</code>	198	Идентификатор языка.
<code>key</code>	200	Ключ (дополнительный параметр выбора).
<code>long_name</code>	200	Длинное (полное) имя компилятора.
<code>priority_adjustment</code>	204	Поправка к приоритету тестирования для данного языка.
<code>short_name</code>	199	Короткое имя компилятора.
<code>src_sfx</code>	201	Суффикс исходных файлов для компилятора.

## Переменная `id`

Имя переменной:	<code>id</code>
Содержится в:	<code>language</code>
Используются:	<code>serve</code> , <code>compile</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает идентификатор языка программирования. Идентификатор является целым числом от 1 и до максимального количества поддерживаемых языков, задаваемого константой `MAX_LANGUAGE` в исходном файле `prepare.c` (значение в стандартной поставке — 31). Никакие два языка программирования не могут иметь одинаковые идентификаторы. Если идентификатор явно не задан, он назначается автоматически. Для этого берётся идентификатор предыдущего языка и увеличивается на 1. Если при этом получится уже использованный идентификатор, диагностируется фатальная ошибка. Если идентификатор первого языка не задан, он устанавливается равным 1.

### Пример.

```
id = 10
```

## Переменная `compile_id`

Имя переменной:	<code>compile_id</code>
Содержится в:	<code>language</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет установить дополнительный идентификатор языка для передачи решения участника на компиляцию. Переменная предназначена для случаев, когда одна и та же программа `compile` обслуживает сразу несколько турниров (см. также конфигурационную переменную `compile_dir`). В этом случае идентификатор языка программирования для сервера компиляции `compile` может отличаться от идентификатора языка сервера турнира `serve`. Данный конфигурационный параметр используется программой `serve` и позволяет задать идентификатор языка, соответствующий идентификатору языка у программы `compile`. Если данная переменная не указана, она полагается равной значению переменной `id`, то есть в случае, если `serve` и `compile` используют один конфигурационный файл, установка переменной `compile_id` не требуется.

### Пример.

```
id = 10
```

## Переменная `disabled`

Имя переменной:	<b>disabled</b>
Содержится в:	<a href="#">language</a>
Используются:	<code>serve</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена в *true*, то участники не могут использовать данный язык программирования для отправки решений. Данный параметр может оказаться полезным, когда по ходу турнира необходимо запретить какой-либо язык программирования. Если просто удалить соответствующую запись, то, во-первых, могут измениться идентификаторы оставшихся языков, что приведёт к проблемам с базой решений участников турнира, и, во-вторых, возникнут проблемы с решениями участников, уже сданных на удалённом языке (вторичный ключ «язык программирования» в базе решений участников турнира будет указывать «в никуда»). Поэтому в таких случаях следует устанавливать конфигурационную переменную `disabled`.

### Пример.

```
disabled
```

## Переменная `short_name`

Имя переменной:	<b>short_name</b>
Содержится в:	<a href="#">language</a>
Используются:	<code>serve</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает «короткое имя» языка программирования. Короткое имя используется в таблице решений участников в CGI-программе `master`, в таблице решений в CGI-программе `team`, а также для выбора языка программирования для отсылки решения. Если эта конфигурационная переменная не установлена, она автоматически устанавливается в значение `lang<id>`, где `<id>` — это идентификатор задачи, преобразованный в строку с помощью спецификации формата `%d`.

### Пример.

```
short_name = "gcc"
```



## Переменная `long_name`

Имя переменной:	<b>long_name</b>
Содержится в:	<a href="#">language</a>
Используются:	<b>serve</b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает «длинное имя» языка программирования. Длинное имя может содержать дополнительную информацию о языке программирования, например, версию компилятора. CGI-программа **team** использует значение этой переменной в меню выбора языка для отсылки решения. Если эта конфигурационная переменная не установлена, она автоматически устанавливается в значение "Language <id>", где <id> — это идентификатор задачи, преобразованный в строку с помощью спецификации формата %d.

### Пример.

```
long_name = "GNU C 3.3.1"
```

## Переменная `key`

Имя переменной:	<b>key</b>
Содержится в:	<a href="#">language</a>
Используются:	<b>serve</b>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Значение по умолчанию:	<i>""</i> (пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает «ключ» для выбора языков программирования, обслуживаемых программой **compile**. Программа **compile** поддерживает опцию командной строки `-k` с помощью которой можно указать ключ языков программирования. Тогда из всех языков программирования, описанных в конфигурационном файле, будут обслуживаться только те, ключ которых совпадает с ключом, заданным в командной строке. В этом случае значение по умолчанию (пустая строка) переменной `key` является значимым, и такой язык программирования будет обслуживаться, только если в командной строке в качестве параметра опции `-k` указана пустая строка. Если опция `-k` не задана, отбор языков по значению переменной `key` не производится, и значение этой переменной может быть произвольным (в частности, переменная может быть не установлена).

### Пример.

```
key = "langs_1"
```

## Переменная `arch`

<b>Имя переменной:</b>	<b><code>arch</code></b>
<b>Содержится в:</b>	<code>language</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>""</code> (пустая строка)
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт «архитектуру», для которой предназначен данный язык программирования. Архитектура является одним из двух параметров, по которому выбирается тестировщик для данной программы. Например, значение архитектуры `dos` может соответствовать компиляторам для DOS (например, Borland C 3.1), а неустановленное значение архитектуры (пустая строка) может соответствовать компиляторам для Linux, которые генерируют статический исполняемый модуль (например, GCC). Процедура запуска скомпилированной программы для этих архитектур существенно отличается, поэтому для каждой из них должен использоваться отдельный тестировщик. Исполняемый файл, полученный в результате компиляции исходного файла, будет направлен на тестирование к тестировщику с соответствующей архитектурой.

### Пример.

```
arch = "dos"
```

## Переменная `src_sfx`

<b>Имя переменной:</b>	<b><code>src_sfx</code></b>
<b>Содержится в:</b>	<code>language</code>
<b>Используются:</b>	<code>serve</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<code>""</code> (пустая строка)
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает суффикс, который должны иметь файлы, подаваемые на компиляцию данному компилятору языка программирования. Например, компилятор GCC требует, чтобы файл программы на языке Си имел суффикс `.c`, а файл программы на языке Си++ имел суффикс `.cpp`, `.cc` или `.C`.

### Пример.

```
src_sfx = ".c"
```

## Переменная `exe_sfx`

Имя переменной:	<code>exe_sfx</code>
Содержится в:	<code>language</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает суффикс исполняемых файлов, получаемых в результате компиляции данным компилятором. Как правило, суффикс исполняемых файлов зависит от архитектуры. Например, исполняемые файлы для Linux обычно не имеют никакого суффикса, а исполняемые файлы для DOS имеют суффикс `.exe`.

### Пример.

```
exe_sfx = ".exe"
```

## Переменная `cmd`

Имя переменной:	<code>cmd</code>
Содержится в:	<code>language</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает программу, которая запускается для компиляции решения участника турнира. Как правило, эта программа не является непосредственно компилятором, а представляет собой скрипт, запускающий компилятор с необходимыми параметрами.

Программа компиляции, задаваемая этой переменной, запускается с двумя параметрами: первый параметр — имя исходного файла программы, второй параметр — имя файла, в который должен быть помещён результат компиляции. Программа запускается с текущим каталогом, установленным на рабочий каталог, в котором находится исходный файл, и должен находиться файл-результат. Стандартный поток вывода и стандартный поток ошибок перенаправляются в файл, который в случае неудачи компиляции будет использоваться как протокол запуска решения пользователя. Программа компиляции должна завершаться с кодом возврата 0, если компиляция завершилась успешно, и с любым другим кодом возврата в случае ошибки при компиляции.

Если значение конфигурационной переменной `cmd` начинается с символа `'/'`, то есть задаёт абсолютный путь к программе, то используется этот путь. В противном случае полный путь к программе компиляции получается добавлением значения данной конфигурационной переменной `cmd` к значению глобальной конфигурационной переменной `script_dir`. Файл программы компиляции должен допускать выполнение, то есть иметь установленным бит `x` прав доступа.

### Пример.

```
cmd = "gcc"
```

## Переменная `compile_dir`

Имя переменной:	<b><code>compile_dir</code></b>
Содержится в:	<code>language</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная позволяет установить каталог обмена программ `serve` и `compile` для данного языка программирования. Если переменная не установлена, используется значение глобальной переменной `compile_dir`. Данная переменная может использоваться, когда необходимо явно назначить сервер компиляции для некоторого языка программирования. При обычном использовании, когда один сервер компиляции поддерживает компиляцию всех определённых в турнире языков программирования, данная конфигурационная переменная не должна использоваться.

## Переменная `compile_real_time_limit`

Имя переменной:	<b><code>compile_real_time_limit</code></b>
Содержится в:	<code>language</code>
Используются:	<code>compile</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная задаёт ограничение астрономического времени на компиляцию программы участника. В случае, если максимальное время компиляции превышено, проверка программы участника завершается со статусом “Check failed” («Проверка не удалась»). Значение 0 означает отсутствие ограничения времени. Если данная конфигурационная переменная не установлена, используется значение глобальной конфигурационной переменной `compile_real_time_limit`.

**Пример.**

```
compile_real_time_limit = 60
```

## Переменная `binary`

Имя переменной:	<b><code>binary</code></b>
Содержится в:	<code>language</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>
Версия <code>ejudge</code> :	начиная с 2.1.27

**Описание.** Данная конфигурационная переменная указывает, что текст послышки с данным языком программирования должен обрабатываться в бинарном режиме. По умолчанию

обработка ведётся в текстовом режиме, то есть программа, принимаемая на проверку, рассматривается как текстовый файл, представляющий собой последовательность ненулевых байтов. Нулевой байт рассматривается как признак конца файла. Таким образом, для текстовой посылки могут использоваться строковые функции языка Си. При приёме посылки в текстовом режиме проводятся дополнительные проверки, что в принимаемом файле отсутствуют нулевые байты. Если обработка посылки ведётся в бинарном режиме, никаких ограничений на содержимое посылки не накладывается.

Бинарный режим приёма посылок должен использоваться для «языков программирования» (а точнее, форматов файлов), которые могут содержать нулевые байты. К таким форматам файлов относятся все архивы (`.tar`, `.tar.gz`, и т. д.), некоторые форматы документов (`.doc`, `.sxw`, и т. д.). Во всех остальных случаях, например, для приёма исходных текстов программ, рекомендуется использовать текстовый режим.

## Переменная `priority_adjustment`

Имя переменной:	<code>priority_adjustment</code>
Содержится в:	<code>language</code>
Используется:	<code>serve</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Значение по умолчанию:	0
Может повторяться:	<i>нет</i>
Версия <code>ejudge</code> :	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет задать поправку к приоритету тестирования для данного языка программирования. Значение поправки будет прибавлено к базовому значению приоритета тестирования (см. описание глобальной конфигурационной переменной `priority_adjustment`) при вычислении приоритета тестирования посылки. Положительное значение поправки уменьшает приоритет, а отрицательное — увеличивает.

### Пример.

```
priority_adjustment = 2
```

## 2.11.7 Параметры тестировщика

Разнообразные архитектурно-зависимые параметры тестирования решений задач описываются в секции тестировщика `tester`. Для каждой пары задача-архитектура должен быть описан тестировщик. Все параметры, относящиеся непосредственно к задаче, описываются в секции `problem`. Ключом для выбора тестировщика являются либо идентификатор задачи, задаваемый с помощью конфигурационной переменной `id`, либо короткое имя задачи, задаваемое с помощью конфигурационной переменной `short_name`. Архитектура определяется конфигурационной переменной `arch` секции описания языка программирования `language`. При этом несколько языков программирования могут иметь одну и ту же архитектуру. Поскольку секция `language` описывает, скорее, не язык программирования, а компилятор языка программирования, то и параметр `arch` описывает архитектуру, для которой генерирует код данный компилятор.

В секции тестировщика поддерживается наследование свойств неабстрактного тестировщика от абстрактного тестировщика аналогично тому, как наследуются свойства в секции задачи. При наследовании свойств в некоторых случаях выполняются форматные подстановки. Допускаются тестировщики по умолчанию, которые предназначены для проверки всех задач, не проверяемых явно заданными тестировщиками, для некоторой архитектуры. В тестировщиках по умолчанию могут использоваться форматные подстановки.

Текущая версия системы `ejudge` поддерживает до 100 неабстрактных тестировщиков одновременно. Каждый тестировщик по умолчанию считается за один тестировщик независимо от количества задач, для которых он предназначен. Ограничение на максимальное количество тестировщиков задаётся константой `MAX_TESTER` в исходном файле `prepare.c` системы `ejudge`. Чтобы изменить это ограничение нужно изменить значение константы и перекомпилировать систему `ejudge`.

Все конфигурационные переменные тестировщика перечислены ниже.

Название	Стр.	Описание
<code>abstract</code>	210	Флаг абстрактного тестировщика.
<code>any</code>	209	Флаг тестировщика по умолчанию.
<code>arch</code>	212	Архитектура тестировщика.
<code>check_cmd</code>	225	Путь к программе проверки ответа.
<code>check_dir</code>	222	Рабочий каталог тестирования.
<code>clear_env</code>	219	Флаг очистки переменных окружения перед запуском тестируемой программы.
<code>errorcode_file</code>	223	Файл, содержащий код завершения тестируемой программы.
<code>error_file</code>	223	Файл, содержащий вывод на стандартный поток ошибок тестируемой программы.
<code>id</code>	206	Идентификатор тестировщика.
<code>is_dos</code>	214	Флаг DOS-формата текста для тестируемых программ.
<code>key</code>	215	Дополнительный параметр выбора — «ключ».
<code>kill_signal</code>	216	Сигнал принудительного завершения тестируемой программы.
<code>max_data_size</code>	218	Максимальный размер данных тестируемой программы.
<i>Продолжение на следующей странице</i>		

Название	Стр.	Описание
<code>max_stack_size</code>	217	Максимальный размер стека тестируемой программы.
<code>max_vm_size</code>	219	Максимальный размер виртуальной памяти тестируемой программы.
<code>name</code>	208	Имя тестировщика.
<code>no_core_dump</code>	215	Флаг запрета дампа памяти при фатальных сигналах у тестируемой программы.
<code>no_redirect</code>	213	Флаг отключения перенаправления у тестируемой программы.
<code>prepare_cmd</code>	224	Команда подготовки тестируемой программы.
<code>priority_adjustment</code>	227	Поправка к приоритету тестирования.
<code>problem</code>	208	Идентификатор тестируемой задачи.
<code>problem_name</code>	209	Имя тестируемой задачи.
<code>run_dir</code>	221	Каталог обмена программ <code>serve</code> и <code>run</code> .
<code>super</code>	211	Абстрактный тестировщик, свойства которого наследуются.
<code>start_cmd</code>	226	Вспомогательная команда для запуска тестируемой программы.
<code>start_env</code>	227	Переменные окружения для тестируемой программы.
<code>time_limit_adjustment</code>	220	Поправка на время тестирования.

## Переменная `id`

**Имя переменной:** `id`  
**Содержится в:** `tester`  
**Используются:** `serve, run`  
**Тип содержимого:** `integer`  
**Может отсутствовать:** `да`  
**Наследуется:** `нет`  
**Может повторяться:** `нет`

**Описание.** Данная конфигурационная переменная устанавливает идентификатор тестировщика. Идентификатор тестировщика, в отличие от идентификатора задачи или языка программирования, не является ключом при поиске, а используется просто для указания элемента массива тестировщиков, в котором хранится описание данного тестировщика. Поэтому в большинстве случаев явное указание идентификатора тестировщика не требуется. Каждый тестировщик должен иметь уникальный идентификатор в пределах от 1 до `MAX_TESTER`. Если идентификатор тестировщика явно не задан, он назначается автоматически. Для этого берётся идентификатор предыдущего тестировщика и увеличивается на 1. Данная процедура автоматического назначения тестировщика может приводить к ошибке повторяющегося идентификатора, которая приведёт к ошибке запуска системы.

Абстрактные тестировщики не могут устанавливать переменную `id`. Если переменная `id` у абстрактного тестировщика установлена, выдаётся сообщение об ошибке. Идентификатор тестировщика не наследуется от абстрактных тестировщиков неабстрактными тестировщиками.

**Пример.**

id = 10



## Переменная `name`

Имя переменной:	<code>name</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт символическое имя тестировщика. Для неабстрактных тестировщиков оно никак не используется. Если имя тестировщика не задано, автоматически устанавливается имя `tst_%d`, если архитектура тестировщика не задана, и `tst_%d_%s`, если архитектура тестировщика задана. Форматное преобразование `%d` применяется к идентификатору тестировщика `id`, а форматное преобразование `%s` применяется к архитектуре тестировщика `arch`.

Для абстрактных тестировщиков имя тестировщика является ключом, по которому находится тестировщик для выполнения операций наследования свойств неабстрактным тестировщиком от данного абстрактного тестировщика. В абстрактных тестировщиках конфигурационная переменная `name` должна быть задана и должна быть уникальна среди всех абстрактных тестировщиков. Естественно, переменная `name` не наследуется.

### Пример.

```
name = "Dos_tester"
```

## Переменная `problem`

Имя переменной:	<code>problem</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>нет</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает идентификатор задачи, тестируемой данным тестировщиком. Если значение переменной установлено, задача с таким идентификатором, задаваемым конфигурационной переменной `id`, должна существовать. Абстрактные тестировщики и тестировщики по умолчанию не могут устанавливать данную конфигурационную переменную. Для прочих тестировщиков для идентификации задачи, тестируемой данным тестировщиком, должна быть задана либо конфигурационная переменная `problem`, либо конфигурационная переменная `problem_name`, но не обе одновременно. Конфигурационная переменная `problem` не наследуется.

### Пример.

```
problem = 10
```

## Переменная `problem_name`

Имя переменной:	<code>problem_name</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>нет</code>
Может повторяться:	<code>нет</code>

**Описание.** Данная конфигурационная переменная устанавливает короткое имя задачи, тестируемой данным тестировщиком. Если значение переменной установлено, задача с таким коротким именем, задаваемым конфигурационной переменной `short_name`, должна существовать. Абстрактные тестировщики и тестировщики по умолчанию не могут устанавливать данную конфигурационную переменную. Для прочих тестировщиков для идентификации задачи, тестируемой данным тестировщиком, должна быть задана либо конфигурационная переменная `problem`, либо конфигурационная переменная `problem_name`, но не обе одновременно. Конфигурационная переменная `problem_name` не наследуется.

### Пример.

```
problem_name = "A"
```

## Переменная `any`

Имя переменной:	<code>any</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>нет</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, данный тестировщик считается тестировщиком по умолчанию. Такой тестировщик используется для тестирования решений всех задач, для которых не задан явный тестировщик. Тестировщик по умолчанию не может устанавливать конфигурационные переменные `problem` и `problem_name`. Тестировщик по умолчанию может иметь абстрактный тестировщик, свойства которого наследуются. Процедура наследования свойств выполняется при тестировании решения задачи, подходящей под данный тестировщик по умолчанию. При наследовании некоторых переменных выполняются форматные подстановки, для которых используются значения конфигурационных переменных описания задачи, соответствующие той неабстрактной задаче, решение которой тестируется. Таким образом, при каждом наследовании свойств значения переменных, зависящие от параметров задачи, будут принимать значения, соответствующие текущей задаче.

Абстрактные тестировщики не могут устанавливать переменную `any`. Значение этой конфигурационной переменной не наследуется.

### Пример.

```
problem_name = "A"
```

## Переменная `abstract`

Имя переменной:	<code>abstract</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>нет</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в `true`, текущее описание тестировщика является описанием абстрактного тестировщика.

Абстрактный тестировщик не имеет идентификатора (то есть не может использовать параметр `id`), не может быть тестировщиком по умолчанию (то есть не может устанавливать параметр `any`), не может устанавливать тестируемую задачу с помощью конфигурационных переменных `problem` или `problem_name`. Абстрактный тестировщик не может наследовать свойства другого абстрактного тестировщика, то есть использование конфигурационной переменной `super` в описании абстрактного тестировщика не допускается.

Описание абстрактного тестировщика должно устанавливать параметр `name`, который используется для идентификации абстрактного тестировщика. Имя абстрактного тестировщика должно быть уникальным среди всех абстрактных тестировщиков.

Описание абстрактного тестировщика устанавливает значения конфигурационных переменных. Неабстрактный тестировщик указывает имя абстрактного тестировщика в конфигурационной переменной `super`, при этом значение некоторой переменной наследуется из абстрактного тестировщика только в том случае, если эта переменная не установлена в самом небабстрактном тестировщике.

Конкретный тестировщик может наследовать следующие конфигурационные переменные из описания абстрактного тестировщика: `arch`, `check_cmd`, `check_dir`, `clear_env`, `errorcode_file`, `error_file`, `is_dos`, `key`, `kill_signal`, `max_data_size`, `max_stack_size`, `max_vm_size`, `no_core_dump`, `no_redirect`, `prepare_cmd`, `run_dir`, `start_cmd`, `start_env`, `time_limit_adjustment`.

При этом при наследовании конфигурационных переменных `check_cmd`, `check_dir`, `errorcode_file`, `error_file`, `prepare_cmd`, `run_dir`, `start_cmd` выполняется **форматная подстановка** (см. раздел 2.11.2).

**Не наследуются** следующие конфигурационные переменные абстрактной задачи: `abstract`, `any`, `id`), `name`, `problem`, `problem_name`, `super`.

**Пример.**

```
abstract
```

## Переменная `super`

Имя переменной:	<code>super</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>нет</i>
Значение по умолчанию:	не установлено
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная определяет имя абстрактного тестировщика, свойства которого наследует данный неабстрактный тестировщик. Значением переменной должно быть имя (см. конфигурационную переменную `name`) абстрактного тестировщика. Если абстрактного тестировщика с таким именем не существует, программы `serve` и `run` не запускаются. Неабстрактная задача может наследовать свойства единственной абстрактной задачи. Если конфигурационная переменная `super` не установлена, никакого наследования свойств не происходит.

**FIXME.** Вроде бы абстрактная задача может наследовать свойства нескольких других абстрактных задач. Надо проверить!

Наследование некоторой конфигурационной переменной заключается в копировании её значения из описания абстрактной задачи в описание неабстрактной задачи. Для некоторых конфигурационных переменных при этом выполняются форматные подстановки. Значение переменной наследуется только тогда, когда значение переменной в неабстрактной задаче не определено. Если неабстрактная задача устанавливает значение некоторой переменной, всегда используется это значение данной переменной.

```
super = "DOS_Tester"
```

## Переменная `arch`

Имя переменной:	<code>arch</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve, run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<code>""</code> (пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт «архитектуру», для которой предназначен данный тестировщик. Архитектура является одним из двух параметров, по которому выбирается тестировщик для тестируемой программы (вторым является идентификатор задачи). Например, значение архитектуры `dos` может соответствовать компиляторам для DOS (например, Borland C 3.1), а неустановленное значение архитектуры (пустая строка) может соответствовать компиляторам для Linux, которые генерируют статический исполняемый модуль (например, GCC). Процедура запуска скомпилированной программы для этих архитектур существенно отличается, поэтому для каждой из них должен использоваться отдельный тестировщик.

Данная конфигурационная переменная соответствует переменной `arch` секции описания языка. Тестировщик с некоторой архитектурой предназначен только для программ, скомпилированных под ту же самую архитектуру. Пустое значение переменной `arch` является значимым, то есть в этом случае тестировщик будет тестировать только программы, скомпилированные компилятором с пустым значением переменной `arch`.

### Пример.

```
arch = "dos"
```

## Переменная `no_redirect`

Имя переменной:	<code>no_redirect</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если данная конфигурационная переменная установлена в *true*, у программы, запускаемой на тестирование данным тестировщиком, не будут перенаправляться стандартные потоки вывода и ошибок. Весь вывод тестируемой программы в эти потоки будет потерян. Если данная переменная не установлена или установлена в *false*, стандартные потоки вывода и ошибок перенаправляются в файлы и затем включаются в протокол тестирования и/или проверяются на соответствие правильному ответу. Стандартный поток ввода в любом случае перенаправляется из устройства `/dev/null`.

Данная переменная предназначена для случая, когда программа запускается не непосредственно, а через какую-нибудь вспомогательную программу (например, эмулятор DOS). В этом случае вывод вспомогательной программы может не представлять интереса и поэтому может игнорироваться.

### Пример.

```
no_redirect
```

## Переменная `is_dos`

Имя переменной:	<code>is_dos</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<code>boolean</code>
Может отсутствовать:	<code>da</code>
Наследуется:	<code>da</code>
Значение по умолчанию:	<code>false</code>
Может повторяться:	<code>нет</code>

**Описание.** Если значение данной конфигурационной переменной равно *true*, программы, тестируемые данным тестировщиком, работают с текстовыми файлами в формате DOS. В таких файлах строки текста завершаются двумя символами `\r`, `\n`. Если значение данной переменной не установлено или установлено в *false*, тестируемые программы работают с текстовыми файлами в формате Unix, в которых строки текста завершаются символом `\n`. Как правило, формат текстового файла, подаваемого на вход тестируемой программе, должен соответствовать требуемому. Например, некоторые стандартные функции языка **Borland Pascal** могут работать некорректно, если программе на вход подаётся текстовый файл в формате Unix, и наоборот, не все программы, написанные для формата Unix, правильно обрабатывают дополнительный символ `\r` в конце строки.

В случае одновременной поддержки языков программирования с текстовыми файлами и в формате DOS, и в формате Unix, тесты для каждой задачи должны быть записаны в текстовом формате Unix, а для тестировщиков, запускающих программы, требующие DOS-формат, должен быть установлен флаг `is_dos`. В этом случае при копировании тестового файла в рабочий каталог будет выполнено его перекодирование в формат DOS. Обратное перекодирование результатов работы программы из формата DOS в формат Unix не выполняется, поэтому проверяющая программа должна корректно обрабатывать оба текстовых формата.

### Пример.

```
is_dos
```

## Переменная `key`

Имя переменной:	<code>key</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code> , <code>run</code>
Тип содержимого:	<i>string</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<code>"</code> (пустая строка)
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает «ключ» для выбора тестируемых, обслуживаемых программой `run`. Программа `run` поддерживает опцию командной строки `-k` с помощью которой можно указать ключ тестируемого. Тогда из всех тестируемых, описанных в конфигурационном файле, будут обслуживаться только те, ключ которых совпадает с ключом, заданным в командной строке. В этом случае значение по умолчанию (пустая строка) переменной `key` является значимым, и такой тестирующий будет обслуживаться, только если в командной строке в качестве параметра опции `-k` указана пустая строка. Если опция `-k` не задана, отбор языков по значению переменной `key` не производится, и значение этой переменной может быть произвольным (в частности, переменная может быть не установлена).

Значение данной конфигурационной переменной наследуется из описания абстрактного тестируемого, если конкретный тестирующий указал имя этого абстрактного тестируемого в переменной `super`, конкретный тестирующий не определяет переменную `key`, а абстрактный — определяет.

### Пример.

```
key = "langs_1"
```

## Переменная `no_core_dump`

Имя переменной:	<code>no_core_dump</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена в значение *true*, при запуске тестируемой программы запрещается генерация дампа памяти (core dump) при аварийном завершении тестируемой программы.



## Переменная `kill_signal`

**Имя переменной:** `kill_signal`  
**Содержится в:** `tester`  
**Используются:** `run`  
**Тип содержимого:** `string`  
**Может отсутствовать:** `da`  
**Наследуется:** `da`  
**Значение по умолчанию:** `term`  
**Может повторяться:** `нет`

**Описание.** Данная конфигурационная переменная устанавливает имя сигнала, который будет послан тестируемой программе при истечении времени её работы, определяемом с помощью конфигурационной переменной `real_time_limit`. Поддерживаются следующие имена сигналов:

Имя в конфигурационном файле	Системное имя сигнала
<code>kill</code>	<code>SIGKILL</code>
<code>term</code>	<code>SIGTERM</code>
<code>int</code>	<code>SIGINT</code>

Текущая версия системы **ejudge** ограничивает время выполнения программы на одном тесте (параметр `time_limit`) с помощью установки ограничения процессорного времени программы с помощью системной функции `setrlimit` (см. также команду интерпретатора **bash** `ulimit`). Такой подход имеет следующие особенности:

1. По истечении добавленной одной секунды тестируемая программа всегда снимается с выполнения посылкой ей сигнала `SIGKILL`.
2. Если тестируемая программа не занимает процессор, например, выполняя системный вызов `pause`, её истраченное процессорное время не увеличивается, поэтому программа может находиться в таком состоянии сколь угодно долго.

Для борьбы с последней особенностью используется ограничение астрономического времени работы программы, задаваемое с помощью конфигурационной переменной `real_time_limit`. Данный параметр `kill_signal` влияет на сигнал, посылаемый тестируемой программе по истечении астрономического времени работы, но не влияет на сигнал, посылаемый программе по истечении процессорного времени работы.

### Пример.

```
kill_signal = "kill"
```

## Переменная `max_stack_size`

Имя переменной:	<code>max_stack_size</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает максимальный размер сегмента стека для тестируемой программы. Размер задаётся в байтах (так, 8388608 байт — это 8 Мегабайт). Если размер стека превышен, программа, как правило, получает фатальный сигнал SIGSEGV (Segmentation fault) и снимается с выполнения. В этом случае тестирование программы завершается с ошибкой “Runtime error”.

Использование данной конфигурационной переменной имеет смысл, если тестируемая программа запускается непосредственно, а не с помощью эмулятора или интерпретатора, так как в этих случаях данная конфигурационная переменная будет ограничивать размер стека эмулятора или интерпретатора.

Если данная конфигурационная переменная не установлена ни в самом неабстрактном тестировщике, ни в абстрактном тестировщике, свойства которого он наследует, ограничение размера стека при запуске программы явно не устанавливается. В этом случае работает ограничение на размер стека, действительное для программы `run` в момент её запуска. Ограничение на размер стека можно просмотреть с помощью команды `ulimit -s` командного интерпретатора `bash`.

### Пример.

```
max_stack_size = 8388608
```

## Переменная `max_data_size`

Имя переменной:	<code>max_data_size</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает максимальный размер сегмента данных для тестируемой программы. Размер задаётся в байтах (так, 8388608 байт — это 8 Мегабайт). К сожалению, современные библиотеки поддержки языка Си выделяют динамическую память таким образом, что ограничение на максимальный размер сегмента данных обходится, поэтому для ограничения максимального размера данных программы приходится использовать конфигурационную переменную `max_vm_size`. Использование данной конфигурационной переменной `max_data_size`, по-видимому, не имеет смысла, и она сохранена для симметрии с `max_stack_size`.

Если данная конфигурационная переменная не установлена ни в самом неабстрактном тестировщике, ни в абстрактном тестировщике, свойства которого он наследует, ограничение размера стека при запуске программы явно не устанавливается. В этом случае работает ограничение на размер стека, действительное для программы `run` в момент её запуска. Ограничение на размер стека можно просмотреть с помощью команды `ulimit -d` командного интерпретатора `bash`.

### Пример.

```
max_data_size = 8388608
```

## Переменная `max_vm_size`

Имя переменной:	<code>max_vm_size</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает максимальный размер виртуальной памяти процесса, исполняющего тестируемую программу. В размере виртуальной памяти учитываются сегмент кода (`.text`) самой программы и всех библиотек, используемых программой, сегмент инициализированных статических данных самой программы и всех используемых библиотек (`.data`), сегмент нулевых статических данных программы и библиотек (`.bss`), динамически растущий сегмент динамической памяти («кучи»), динамически растущий сегмент стека. Размер задаётся в байтах (так, 16777216 байт — это 16 Мегабайт).

Если данная конфигурационная переменная не установлена ни в самом неабстрактном тестировщике, ни в абстрактном тестировщике, свойства которого он наследует, ограничение размера стека при запуске программы явно не устанавливается. В этом случае работает ограничение на размер стека, действительное для программы `run` в момент её запуска. Ограничение на размер стека можно просмотреть с помощью команды `ulimit -v` командного интерпретатора `bash`.

### Пример.

```
max_vm_size = 16777216
```

## Переменная `clear_env`

Имя переменной:	<code>clear_env</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>boolean</i>
Может отсутствовать:	<i>da</i>
Наследуется:	<i>da</i>
Значение по умолчанию:	<i>false</i>
Может повторяться:	<i>нет</i>

**Описание.** Если данная конфигурационная переменная установлена в *true*, при запуске тестируемой программы на выполнение сбрасываются все переменные окружения, и тестируемая программа получает пустое окружение. После сброса всех переменных можно установить новые переменные окружения с помощью конфигурационной переменной `start_env`.

### Пример.

```
clear_env
```

## Переменная `time_limit_adjustment`

Имя переменной:	<code>time_limit_adjustment</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>integer</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<code>0</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает «поправку» к ограничению времени тестирования на одном тесте для всех программ, тестируемых данным тестировщиком. Данная переменная задаёт дополнительное время в секундах и должна быть не меньше нуля. Ограничение времени тестирования на одном тесте определяется как сумма значения переменной `time_limit` описания задачи и значения данной конфигурационной переменной `time_limit_adjustment`. Если значение переменной `time_limit` не установлено, то и переменная `time_limit_adjustment` не используется.

Данная конфигурационная переменная может оказаться полезной, когда тестируемая программа запускается не непосредственно, а из под эмулятора. Тогда во времени тестирования необходимо учесть время старта самого эмулятора. Например, для `dosemu` это время составляет примерно 0.5 сек. Секундная точность поправки может казаться недостаточной, но, к сожалению, таково ограничение операционной системы (ограничение на процессорное время задаётся с секундной точностью).

### **Пример.**

```
time_limit_adjustment = 1
```

## Переменная `run_dir`

Имя переменной:	<code>run_dir</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная задаёт путь к каталогам обмена между программами `serve` и `run`. Она используется только программой `serve`, и позволяет для некоторых тестировщиков переопределять путь к каталогу обмена с программой `run`, устанавливаемый глобальной переменной `run_dir`. Каждая программа `run` может иметь единственный каталог обмена, который она, однако, может разделять с другими программами `run`. Если одновременно несколько программ `run` (например, находящихся на разных компьютерах в локальной сети) используют один и тот же каталог обмена, расположенный на сетевом диске, запросы на тестирование будут распределяться между ними случайно. Но для того, чтобы часть запросов на тестирование программой `serve` отдавались одному процессу (или группе процессов, использующих один и тот же каталог обмена), а часть — другому, может использоваться данная конфигурационная переменная.

**BUG.** Текущая версия системы `ejudge` никак не использует данную конфигурационную переменную.

### Пример.

```
run_dir = /var/ejudge/run/var/compile
```

## Переменная `check_dir`

Имя переменной:	<code>check_dir</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к каталогу</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к рабочему каталогу, в котором тестируется решение участника. В этот каталог копируется исполняемая программа и входные данные для неё, в нём же создаются выходные файлы нулевого размера. Решение участника запускается с данным каталогом установленным в качестве текущего каталога. После каждого запуска этот каталог полностью очищается. Для того, чтобы ограничить максимальный размер выходных файлов, создаваемых программой участника, рекомендуется разместить этот каталог на файловой системе ограниченного размера, отдельной от основных файловых систем. Для этого можно использовать, например, loopback-устройство, указав в качестве флагов монтирования `-o loop`.

Если данная переменная не установлена в описании неабстрактного тестировщика, но абстрактный тестировщик, указанный в переменной `super`, устанавливает данную переменную, используется значение, указанное в абстрактном тестировщике, при этом выполняются [форматные подстановки](#). Если после этого значение данной переменной всё равно не определено, используется значение глобальной конфигурационной переменной `run_check_dir`.

**Пример.** Для создания небольшой файловой системы, монтируемой с помощью устройства-петли (loopback), можно воспользоваться следующей последовательностью команд.

- Создаём файл требуемого размера (в нашем случае — 32 Мб).

```
dd if=/dev/zero of=/var/ejudge/image bs=1M count=32
```

- Создаём на нём файловую систему.

```
mke2fs -f /var/ejudge/image
```

- Создаём каталог монтирования.

```
mkdir /var/ejudge/disk
```

- Монтируем новую файловую систему.

```
mount /var/ejudge/image /var/ejudge/disk -o loop
```

- Создаём в нём нужный каталог и устанавливаем его права.

```
mkdir /var/ejudge/disk/work
chown user:user /var/ejudge/disk/work
chmod 755 /var/ejudge/disk/work
```

Здесь `user` — пользователь, из-под которого будет проводиться тестирование решений.

После этого в конфигурационном файле турнира `serve.cfg` можно установить глобальную конфигурационную переменную `run_check_dir` следующим образом:

```
run_check_dir = /var/ejudge/disk/work
```

## Переменная `errorcode_file`

Имя переменной:	<b><code>errorcode_file</code></b>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>имя файла</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает имя файла, в котором после завершения выполнения тестируемой программы находится код завершения программы. В этом случае код завершения процесса программы решения участника игнорируется программой `run`. Эта переменная может оказаться полезной, когда тестируемая программа участника запускается не непосредственно, а из-под эмулятора, который не обеспечивает передачу в вызвавшую эмулятор программу кода завершения эмулируемой программы. Эмулятор DOS `dosemu` — это пример такого эмулятора. Естественно, в случае использования файла кода завершения решение участника должно запускаться специальной программой (также работающей под эмулятором), которая обеспечит помещение кода завершения программы в требуемый файл.

Если переменная `errorcode_file` установлена, файл с таким именем будет размещаться в рабочем каталоге тестирования, задаваемом переменной `check_dir`. Значение переменной `errorcode_file` не должно содержать символа `'/'`, то есть должно быть именем файла в текущем каталоге.

### Пример.

```
errorcode_file = error.out
```

## Переменная `error_file`

Имя переменной:	<b><code>error_file</code></b>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>имя файла</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<code>error</code>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает имя файла, в который перенаправляется вывод на стандартный поток ошибок тестируемой программы. После окончания тестирования содержимое этого файла вставляется в протокол тестирования. Переменная не используется (то есть перенаправления не происходит), если конфигурационная переменная `no_redirect` установлена в значение `true`. Файл перенаправления потока ошибок создаётся в рабочем каталоге тестирования, задаваемом конфигурационной переменной `check_dir`. Значение переменной `error_file` не должно содержать символа `'/'`, то есть должно быть именем файла в текущем каталоге.



## Переменная `prepare_cmd`

Имя переменной:	<code>prepare_cmd</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к программе, которая подготавливает выполнение тестируемой программы. В качестве параметра подготавливающей программе передаётся имя исполняемого тестируемого файла (без полного пути). Подготавливающая программа запускается в каталоге временных файлов тестировщика, задаваемом глобальной конфигурационной переменной `run_work_dir`. Её вывод добавляется в протокол тестирования. Подготавливающая программа должна завершаться с кодом возврата 0, или в противном случае тестирование завершится ошибкой “Check Failed”. Подготавливающая программа запускается один раз перед тестированием программы, а не перед каждым запуском программы на очередном тесте.

Если данная переменная в описании тестировщика не определена, но абстрактный тестировщик, указанный в переменной `super` определяет эту переменную, используется значение из абстрактного тестировщика, при этом выполняются [форматные подстановки](#). Неопределённое значение переменной допускается, и в этом случае подготавливающая программа не запускается. Если значение данной переменной не начинается с `'/'`, то есть является относительным путём, оно добавляется к значению глобальной конфигурационной переменной `script_dir`.

**FIXME.** Нужно, чтобы в программу подготовки передавались бы более вразумительные параметры. Текущий каталог должен устанавливаться, по-видимому, в каталог тестирования `check_dir`.

### Пример.

```
prepare_cmd = "set_perms"
```

## Переменная `check_cmd`

Имя переменной:	<code>check_cmd</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>нет</i>
Наследуется:	<i>да</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к проверяющей программе, которая запускается каждый раз после завершения работы тестируемой программы на очередном тесте и проверяет правильность ответа тестируемой программы. Проверяющая программа запускается только если тестируемая программа уложилась в отведённое на работу над одним тестом время (см. переменную `time_limit`) и завершилась успешно, то есть выработав код возврата 0. Проверяющей программе передаются два или три параметра в зависимости от того, установлена ли переменная `use_corr` описания задачи, то есть используется ли при тестировании решения заранее заготовленный файл с правильным ответом.

- Первый параметр (`argv[1]`) проверяющей программы — путь к файлу, в котором находятся входные данные текущего теста. Передаётся путь к файлу, находящемуся в тестовом каталоге (см. переменную `test_dir`), а не к файлу, скопированному в каталог тестирования перед запуском тестируемой программы. Таким образом, вне зависимости от значения переменной `is_dos` проверяющая программа работает с непреобразованным входным файлом.
- Второй параметр (`argv[2]`) проверяющей программы — путь к файлу, в котором находится результат работы тестируемой программы. Преобразования формата файла не выполняется, поэтому проверяющая программа должна обрабатывать концы строк в стиле DOS (`'\r'`, `'\n'`) для проверки решения DOS- или win32-программ, и в стиле Unix (`'\n'`) для проверки решения Linux-программ.
- Если установлена конфигурационная переменная `use_corr`, третий параметр (`argv[3]`) проверяющей программы — путь к файлу, содержащему правильный ответ на текущий тест (см. переменную `corr_dir`). Если конфигурационная переменная `use_corr` не установлена, третий параметр в проверяющую программу не передаётся.

Проверяющая программа запускается с текущим каталогом, установленным в рабочий каталог тестирования. Вывод проверяющей программы на стандартный поток вывода или стандартный поток ошибок сохраняется и добавляется в файл протокола. Результат тестирования определяется по коду возврата, выработанному проверяющей программой.

0	OK — тестируемая программа выдала верный ответ.
4	“Presentation error”
5	“Wrong answer”

Если проверяющая программа выработала любой другой код завершения, завершилась из-за прихода сигнала (любого), или превысила отведённый ей лимит времени (см. переменную `checker_real_time_limit`), тестирование программы участника завершается со статусом “Check failed”.

Если данная конфигурационная переменная не установлена, но абстрактный тестировщик, указанный в переменной `super`, устанавливает эту переменную, используется значение переменной из описания абстрактного тестировщика, при этом выполняются **форматные подстановки**. Если после этого значение данной переменной `check_cmd` всё ещё не определено, программа `run` выдаёт ошибку и отказывается запускаться. Если значение переменной `check_cmd` не начинается с символа `'/'`, то есть является относительным путём, оно добавляется к значению глобальной конфигурационной переменной `checker_dir`.

#### Пример.

```
check_cmd = "check_a"
```

### Переменная `start_cmd`

Имя переменной:	<code>start_cmd</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<i>путь к файлу</i>
Может отсутствовать:	<i>да</i>
Наследуется:	<i>да</i>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<i>нет</i>

**Описание.** Данная конфигурационная переменная устанавливает путь к вспомогательной программе, используемой для запуска тестируемой программы участника. Эта вспомогательная программа должна выполнить необходимые подготовительные операции, а затем вызвать тестируемую программу с помощью системного вызова `execv` или `execve`. Полный путь к тестируемой программе передаётся первым аргументом командной строки, а дополнительные аргументы для тестируемой программы — последующими параметрами. Простейший способ вызова тестируемой программы из вспомогательной программы заключается в вызове функции `execv` со следующими параметрами.

```
execv(argv[1], argv + 1);
```

Идентификатор процесса тестируемой программы должен быть равен идентификатору процесса вспомогательной программы (то есть не допускается использование `fork`, `system`, `popen` и т. д.), так как в противном случае не гарантируется корректное завершение тестируемой программы по истечению лимита времени.

Если данная конфигурационная переменная не установлена, но абстрактный тестировщик, указанный в переменной `super`, устанавливает эту переменную, используется значение переменной из описания абстрактного тестировщика, при этом выполняются **форматные подстановки**. Если значение переменной `start_cmd` не начинается с символа `'/'`, то есть является относительным путём, оно добавляется к значению глобальной конфигурационной переменной `script_dir`. Если после всех указанных выше действий значение переменной `start_cmd` не определено, никакая вспомогательная программа запускаться не будет, а программа `run` будет запускать тестируемую программу непосредственно.

#### Пример.

```
start_cmd = "capexec"
```

## Переменная `start_env`

Имя переменной:	<code>start_env</code>
Содержится в:	<code>tester</code>
Используются:	<code>run</code>
Тип содержимого:	<code>string</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<i>не установлено</i>
Может повторяться:	<code>да</code>

**Описание.** Данная конфигурационная переменная позволяет задавать переменные окружения, передаваемые в тестируемую программу. Каждая спецификация переменной окружения имеет вид

```
NAME=VALUE
```

С помощью конфигурационной переменной `start_env` можно устанавливать по одной переменной окружения за раз. Чтобы установить несколько переменных окружения, конфигурационная переменная `start_env` должна использоваться требуемое число раз.

Если и абстрактный тестировщик, указанный в переменной `super` для данного тестировщика, и сам тестировщик устанавливают переменные окружения, спецификации переменных окружения объединяются так, что переменные окружения из абстрактного тестировщика идут первыми, а за ними — переменные окружения из неабстрактного тестировщика.

### Пример.

```
clear_env
start_env = "LD_BIND_NOW=1"
start_env = "LD_PRELOAD=/usr/lib/ejudge/libdropcaps.so"
```

## Переменная `priority_adjustment`

Имя переменной:	<code>priority_adjustment</code>
Содержится в:	<code>tester</code>
Используются:	<code>serve</code>
Тип содержимого:	<code>integer</code>
Может отсутствовать:	<code>да</code>
Наследуется:	<code>да</code>
Значение по умолчанию:	<code>0</code>
Может повторяться:	<i>нет</i>
Версия <b>ejudge</b> :	начиная с 2.1.29

**Описание.** Данная конфигурационная переменная позволяет задать поправку к приоритету тестирования для данного тестировщика. Значение поправки будет прибавлено к базовому значению приоритета тестирования (см. описание глобальной конфигурационной переменной `priority_adjustment`) при вычислении приоритета тестирования посылки. Положительное значение поправки уменьшает приоритет, а отрицательное — увеличивает.

### Пример.

```
priority_adjustment = 2
```

## 2.12 Файл описания теста `test.inf`

В данном разделе описывается формат файла описания теста. В нем определяются такие характеристики теста, как аргументы командной строки, передаваемые тестируемой программе, комментарий к тесту, комментарий для участников (подсказка) к тесту.

### 2.12.1 Использование

Файлы описания тестов используются только в случае, если конфигурационная переменная `use_info` описания задачи установлена в `true`. Для каждой задачи турнира этот флаг может устанавливаться независимо от других задач. Если конфигурационная переменная `use_info` установлена, то для каждого теста, определённого для задачи, должен существовать файл описания теста. Эти файлы располагаются в каталоге, задаваемом конфигурационной переменной `info_dir` описания задачи. Каждый файл описания теста должен называться `NUM.SFX`, где `NUM` — номер теста, которому он соответствует, напечатанный тремя цифрами, включая ведущие нули. `SFX` — это суффикс файлов описания тестов, который по умолчанию равен `.inf`, но может быть переопределён с помощью конфигурационной переменной `info_sfx`. Например, если для некоторой задачи определены 3 теста и установлен флаг `use_info`, то в каталоге, определяемом переменной `info_dir`, должны находиться файлы `001.inf`, `002.inf`, `003.inf` (при условии, что используется значение переменной `info_sfx` по умолчанию).

Далее для удобства изложения файл описания теста мы будем называть `test.inf`.

### 2.12.2 Общая структура

Файл `test.inf` представляет собой текстовый файл, аналогичный по структуре файлу `serve.cfg`. Файл имеет построчную структуру. Строки текста в файле, состоящие только из пробельных символов, игнорируются. Содержимое строки от первого в ней символа `#` и до конца строки игнорируется.

Значимая строка в конфигурационном файле имеет следующий формат:

```
NAME = VALUE
```

Здесь `NAME` — это имя конфигурационной переменной. Имя конфигурационной переменной может состоять из латинских заглавных и строчных букв, цифр и символа подчёркивания. Все поддерживаемые конфигурационные переменные описаны ниже. `VALUE` — это значение переменной. Допускается строка вида

```
NAME
```

В этом случае `VALUE` полагается равным 1.

`VALUE` может состоять из нескольких «слов», аналогично тому, как команда интерпретатора строки состоит из имени команды и её аргументов. К `VALUE` применяется процедура разбиения строки на слова, аналогичная процедуре, реализованной в командном интерпретаторе `bash`. Пробельные символы в начале и конце значения `VALUE` отбрасываются. Слова в `VALUE` — это последовательности символов отделённые друг от друга произвольным числом пробельных символов. При разбиении на слова учитываются специальные символы `'`, `"` и `\`.

Символ ' начинает последовательность символов, которая завершается первым символом '. В этой последовательности пробельные символы рассматриваются как обычные символы, и символы ", \ не работают как специальные. После чтения такой последовательности символы ' с концов отбрасываются.

Символ " начинает последовательность символов, которая завершается первым символом ". В этой последовательности пробельные символы рассматриваются как обычные символы, и символы ', \ не работают как специальные. После чтения такой последовательности символы " с концов отбрасываются.

Символ \ воздействует на следующий за ним символ. Для некоторых символов (пробельные, ', ") отменяется их специальный смысл. Для других символов (n, r) они интерпретируются как специальная запись управляющего кода (как в языке Си). Кроме этого можно задавать код символа в восьмеричной или шестнадцатеричной записи (как в языке Си). Полное описание поддерживаемых последовательностей дано в таблице 2.14. Если подходят одновременно несколько вариантов из данной таблицы, предпочтение отдаётся варианту максимальной длины.

<code>\ X H<sub>1</sub>H<sub>2</sub></code>	Задаёт символ с указанным шестнадцатеричным кодом. Здесь <i>X</i> — символ <i>x</i> или <i>X</i> , <i>H<sub>1</sub></i> , <i>H<sub>2</sub></i> — шестнадцатеричные цифры ([0–9a–fA–F]).
<code>\ X H<sub>1</sub></code>	Задаёт символ с указанным шестнадцатеричным кодом. <i>H<sub>1</sub></i> — шестнадцатеричная цифра.
<code>\ O<sub>1</sub>O<sub>2</sub>O<sub>3</sub></code>	Задаёт символ с указанным восьмеричным кодом. Здесь <i>O<sub>1</sub></i> — цифра от 0 до 3, <i>O<sub>2</sub></i> и <i>O<sub>3</sub></i> — восьмеричные цифры ([0–7]).
<code>\ O<sub>1</sub>O<sub>2</sub></code>	Задаёт символ с указанным восьмеричным кодом. <i>O<sub>1</sub></i> , <i>O<sub>2</sub></i> — восьмеричные цифры.
<code>\ O<sub>1</sub></code>	Задаёт символ с указанным восьмеричным кодом. <i>O<sub>1</sub></i> — восьмеричная цифра.
<code>\ a</code>	Символ <code>\a</code> в нотации языка Си.
<code>\ b</code>	Символ <code>\b</code> в нотации языка Си.
<code>\ f</code>	Символ <code>\f</code> в нотации языка Си.
<code>\ n</code>	Символ <code>\n</code> в нотации языка Си.
<code>\ r</code>	Символ <code>\r</code> в нотации языка Си.
<code>\ t</code>	Символ <code>\t</code> в нотации языка Си.
<code>\ v</code>	Символ <code>\v</code> в нотации языка Си.
<code>\ C</code>	Символ <i>C</i> , здесь <i>C</i> — символ, не перечисленный выше.

Таблица 2.14: Действие символа обратной косой черты

### Примеры.

*Пример 1.*

`_____a__b_____c`

В данном случае строка будет разбита на три слова: *a*, *b* и *c*.

*Пример 2.*

`"__a_"\_'__b_'_'_"aa"__'`

В данном случае строка будет разбита на два слова: `__a____b_` и `_"aa"__'`.

### 2.12.3 Конфигурационные переменные

В файле описания теста `test.inf` допускаются следующие конфигурационные переменные:

Название	Стр.	Описание
<code>comment</code>	230	Основной (судейский) комментарий к тесту.
<code>params</code>	231	Параметры командной строки для тестируемой программы.
<code>team_comment</code>	230	Комментарий к тесту для участников.

#### Переменная `comment`

**Имя переменной:** `comment`  
**Используются:** `run`  
**Тип содержимого:** *string*  
**Может отсутствовать:** *да*  
**Значение по умолчанию:** *не установлено*  
**Может повторяться:** *нет*

**Описание.** Данная конфигурационная переменная позволяет задать комментарий к тесту. Текст комментария вставляется в судейский протокол тестирования и недоступен участникам турнира. Значение данной конфигурационной переменной должно быть одним словом.

##### Пример.

```
comment = "Тест на отрицательные X"
```

#### Переменная `team_comment`

**Имя переменной:** `team_comment`  
**Используются:** `run`  
**Тип содержимого:** *string*  
**Может отсутствовать:** *да*  
**Значение по умолчанию:** *не установлено*  
**Может повторяться:** *нет*

**Описание.** Данная конфигурационная переменная позволяет задать комментарий к тесту для участников. Текст этого комментария вставляется в протокол тестирования, доступный команде-автору решения, если на данном тесте решение команды дало неправильный результат. Значение данной конфигурационной переменной должно быть одним словом.

##### Пример.

```
team_comment = "Проверьте диапазон переменной X!"
```

## Переменная `params`

<b>Имя переменной:</b>	<b><code>params</code></b>
<b>Используются:</b>	<code>run</code>
<b>Тип содержимого:</b>	<i>string</i>
<b>Может отсутствовать:</b>	<i>да</i>
<b>Значение по умолчанию:</b>	<i>не установлено</i>
<b>Может повторяться:</b>	<i>нет</i>

**Описание.** Данная конфигурационная переменная позволяет установить параметры командной строки для тестируемой программы. Значение данной переменной может состоять из нескольких слов. Каждое слово будет отдельным параметром командной строки.

### **Пример.**

```
params = a b c
```

В данном случае тестируемой программе будут переданы три параметра командной строки: `a`, `b`, `c`. Как обычно, нулевым параметром командной строки будет передан путь к самой запускаемой программе. Таким образом, общее количество параметров (значение аргумента `argc` в функции `main`) окажется равным 4.



# Глава 3

## Разработка проверяющих программ

В данной главе описывается интерфейс, которому должна удовлетворять проверяющая программа, работающая в системе `ejudge`. Далее описывается библиотека `libchecker` для проверяющих программ на языке Си, которая призвана упростить написание проверяющих программ.

### 3.1 Проверяющие программы

Проверяющая программа — это программа, которая используется для сравнения ответа, который был выдан программой участника турнира на некотором тесте, с правильным ответом на этом тесте. Для своей работы проверяющая программа может использовать входной тестовый файл, файл, содержащий результат работы тестируемой программы, и файл с правильным ответом (либо с любой другой информацией, которая может быть использована для проверки правильности ответа).

Проверяющая программа запускается только в тех случаях, когда программа участника уложились в отведённое ей время и завершилась успешно, то есть с кодом возврата 0. В остальных случаях диагностируется превышение времени выполнения или ошибка периода выполнения соответственно, и проверяющая программа не запускается.

#### 3.1.1 Параметры командной строки

В зависимости от значения конфигурационных переменных `use_corr`, `use_info` и `use_tgz` (доступных как на глобальном уровне, так и на уровне описания задачи) проверяющая программа может получать от 2 до 6 параметров командной строки.

Первый параметр (`argv[1]`) командной строки — это всегда путь к файлу с тестовыми данными. Несмотря на то, что каждый раз файл с тестовыми данными копируется из каталога тестов в рабочий каталог тестирования, проверяющей программе всегда передаётся путь к файлу с тестовыми данными, находящемуся в каталоге тестов, а не в рабочем каталоге тестирования, так как тестируемая программа может случайно или намеренно испортить свой входной файл.

Второй параметр (`argv[2]`) командной строки проверяющей программы — это всегда путь к файлу с результатом работы тестируемой программы. Этот файл находится в рабочем каталоге тестирования.

Все последующие параметры командной строки являются необязательными. Их наличие или отсутствие зависят от значения перечисленных выше конфигурационных переменных.

Если значение конфигурационной переменной `use_corr` установлено в `true`, то следующим параметром командной строки (`argv[3]`) передаётся путь к файлу с правильным ответом на данный тест. Этот файл находится в каталоге правильных ответов. На самом деле, этот файл не обязан содержать в точности правильный ответ, но может содержать произвольную информацию, которая поможет проверяющей программе проверить правильность ответа тестируемой программы.

Если значение конфигурационной переменной `use_info` установлено в `true`, то следующим параметром командной строки (`argv[3]` или `argv[4]` в зависимости от значения переменной `use_corr`) передаётся путь к файлу с дополнительной информацией о тесте (см. раздел 2.12).

Наконец, если значение конфигурационной переменной `use_tgz` установлено в `true`, проверяющей программе передаются ещё два параметра командной строки: путь к незаархивированному исходному содержимому рабочего каталога тестируемой программы и путь к рабочему каталогу тестируемой программы. Проверяющая программа может сравнивать первый каталог со вторым, чтобы определить, как изменился рабочий каталог в результате работы программы.

### 3.1.2 Результат проверки

Проверяющая программа может выводить произвольную информацию на стандартный поток вывода и стандартный поток ошибок. Вся выведенная информация будет добавлена в протокол тестирования.

Проверяющая программа сообщает системе `ejudge` свой вердикт с помощью кода завершения программы.

Код завершения	Вердикт
0	ОК — тестируемая программа на данном тесте дала правильный ответ.
4	Presentation error — результат, выведенный тестируемой программой, не может быть проверен на правильность, поскольку нарушен специфицированный в условии задачи формат вывода результата.
5	Wrong answer — тестируемая программа выработала неправильный ответ.
6	Checker error — внутренняя ошибка проверяющей программы.

Если проверяющая программа завершается аварийно из-за получения фатального сигнала, или код её завершения не равен одному из кодов, перечисленных выше, система `ejudge` считает, что произошла внутренняя ошибка проверяющей программы. Такие послышки отмечаются в базе данных посылок со статусом “Manual check required”.

## 3.2 Библиотека `libchecker`

Библиотека предназначена для разработки проверяющих программ для системы `ejudge`. Задача проверяющей программы — по результату работы тестируемой программы определить, правильный ли ответ был получен тестируемой программой.

Библиотека представляет собой каркас (framework), в который вставляются необходимые действия по проверке результата. В библиотеке определяется функция `main`, то есть при старте проверяющей программы управление получает модуль инициализации библиотеки `libchecker`. После инициализации библиотеки вызывается функция `checker_main`, которая играет роль функции `main` для проверяющей программы.

### 3.2.1 Вид проверяющей программы

Проверяющая программа с использованием библиотеки `libchecker` выглядит следующим образом:

```
1 #define NEED_CORR 0
2 #define NEED_INFO 0
3 #define NEED_TGZ 0
4 #include "checker.h"
5
6 int checker_main(int argc, char **argv)
7 {
8     // checker code
9     checker_OK();
10 }
```

В строках 1, 2 и 3 устанавливается режим работы проверяющей программы. Перед подключением заголовочного файла `"checker.h"` обязательно должны быть определены три макроса `NEED_CORR`, `NEED_INFO`, `NEED_TGZ`.

Макрос `NEED_CORR` необходимо установить в ненулевое значение, если для проверки решения проверяющей программе требуется файл с правильным ответом. Тогда файл с правильным ответом будет открыт автоматически при старте проверяющей программы, и он будет доступен через глобальные переменные `f_corr` и `f_arr[2]`. Путь к файлу с правильным ответом передаётся третьим параметром командной строки и может быть использован и в функции `checker_main`. Обратите внимание, что чтобы программа `run` передавала проверяющей программе путь к файлу с правильным ответом, в разделе описания задачи должна быть установлена конфигурационная переменная `use_corr`.

Если макрос `NEED_CORR` установлен в 0, считается, что файл с правильным ответом для проверяющей программы не нужен, и он не открывается.

Макрос `NEED_INFO` необходимо установить в ненулевое значение, если для проверки решения проверяющей программе необходим файл с дополнительной информацией о тесте `test.inf`. Тогда файл с информацией о тесте будет считан автоматически при старте проверяющей программы, и информация из него будет доступна через глобальную переменную `test_info`. Путь к файлу информации о тесте передаётся в параметрах командной строки и доступен в функции `checker_main`. Чтобы программа `run` передавала проверяющей программе путь к файлу с правильным ответом, в разделе описания задачи должна быть установлена конфигурационная переменная `use_info`.

Если макрос `NEED_INFO` установлен в 0, считается, что файл с информацией о тесте для проверяющей программы не нужен, и он не открывается.

Макрос `NEED_TGZ` необходимо установить в ненулевое значение, если тестируемая программа запускается в специально создаваемом каталоге, начальное содержимое которого копируется из каталога тестов (см. описание конфигурационной переменной `use_tgz`). В таком случае при старте проверяющей программы будет открыт рабочий каталог тестируемой программы и открыт эталонный каталог из тестового каталога. Каталоги открывают-

ся с помощью стандартной функции `opendir`. Дескрипторы открытых каталогов помещаются в глобальные переменные `dir_in` (эталонный каталог), `dir_out` (рабочий каталог после завершения тестируемой программы). Дополнительно в глобальные переменные `dir_in_path`, `dir_out_path` помещаются пути к этим каталогам. Пути к этим каталогам передаются тестирующей программе в аргументах командной строки, но только если конфигурационная переменная `use_tgz` для данной задачи установлена в `true`.

При запуске проверяющей программы будет выполнена начальная инициализация в соответствии с установками, заданными макросами `USE_CORR`, `USE_INFO` и `USE_TGZ`. Затем управление будет передано на функцию `checker_main`, которая должна выполнить проверку и вернуть код возврата в соответствии с соглашениями об интерфейсе проверяющих программ. Функции `checker_main` передаются все аргументы командной строки проверяющей программы.

### 3.2.2 Константы

```
enum
{
    RUN_OK                = 0,
    RUN_PRESENTATION_ERR = 4,
    RUN_WRONG_ANSWER_ERR = 5,
    RUN_CHECK_FAILED      = 6
};
```

Данные константы перечисляют коды завершения, доступные для проверяющей программы. Если проверяющая программа вернёт какой-либо код возврата, кроме указанных выше, программа `run` преобразует этот код возврата в код возврата `RUN_CHECK_FAILED`.

### 3.2.3 Глобальные переменные

В данном разделе перечислены глобальные переменные, определяемые в библиотеке `libchecker` и доступные для использования проверяющими программами.

#### Переменная `f_in`

```
extern FILE *f_in;
```

С помощью данного дескриптора потока можно работать с входным файлом для тестируемой программы, то есть файлом, содержащим тест. Даже если согласно условию задачи тестируемая программа должна считывать данные со стандартного потока ввода, данный дескриптор потока будет связан с файлом входных данных на диске. Входной файл всегда открывается в каталоге тестов, а не в рабочем каталоге программы. Если при открытии входного файла при запуске проверяющей программы возникает ошибка, то выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Функция `checker_main` в этом случае даже не получает управление.

#### Переменная `f_team`

```
extern FILE *f_team;
```

Данная переменная содержит дескриптор потока, связанный с выходным файлом тестируемой программы. Если согласно условию задачи тестируемая программа должна вывести результат на стандартный поток вывода, этот дескриптор потока связан с временным файлом на диске, где находится вся выдача на `stdout` (вывод в `stderr` перенаправляется в другой файл и проверяющей программе не доступен). Если согласно условию задачи результаты должны быть записаны в файл, то этот дескриптор потока связан с этим файлом. Если при открытии выходного файла при запуске проверяющей программы возникает ошибка, то выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_PRESENTATION_ERR`. Функция `checker_main` в этом случае даже не получает управление.

### Переменная `f_corr`

```
extern FILE *f_corr;
```

Данная переменная содержит дескриптор потока, связанный с эталонным файлом (файлом с правильным ответом). Этот файл открывается, если макрос `USE_CORR` установлен в ненулевое значение. Если при этом в проверяющую программу передано неверное количество параметров командной строки, или при открытии файла возникла ошибка, выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Функция `checker_main` в этом случае даже не получает управление.

### Переменная `f_arr`

```
extern FILE *f_arr[3];
```

Данный массив содержит те же самые дескрипторы потоков, что и описанные выше глобальные переменные и введён для удобства программиста. `f_arr[0]` соответствует `f_in`, `f_arr[1]` соответствует `f_team`, а `f_arr[2]` — `f_corr` (если последний открыт).

Некоторые функции требуют в качестве параметра, задающего входной поток, целое число. Это целое число используется как индекс в массиве `f_arr`, то есть 0 — входной файл, 1 — выходной файл тестируемой программы, 2 — эталонный файл.

### Переменная `f_arr_names`

```
extern const unsigned char * const f_arr_names[3];
```

Этот массив содержит «имена» потоков для массива `f_arr`. В соответствующем Си-файле массив определяется следующим образом:

```
const unsigned char * const f_arr_names[3] =  
{  
    "input",  
    "team_output",  
    "correct_output"  
};
```

## Переменная `dir_in`

```
extern DIR *dir_in;
```

Данная переменная содержит дескриптор открытого эталонного рабочего каталога тестируемой программы. Переменная доступна, только если макрос `USE_TGZ` установлен в ненулевое значение. Эталонный рабочий каталог находится в том же каталоге, где и тесты. Эталонный рабочий каталог можно считать специальной формой входных данных программы. Если в проверяющую программу передано неверное количество параметров командной строки, или при открытии каталога возникла ошибка, выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Функция `checker_main` в этом случае даже не получает управление.

## Переменная `dir_out`

```
extern DIR *dir_out;
```

Данная переменная содержит дескриптор открытого рабочего каталога тестируемой программы, оставшегося после её завершения. Переменная доступна, только если макрос `USE_TGZ` установлен в ненулевое значение. Если в проверяющую программу передано неверное количество параметров командной строки, или при открытии каталога возникла ошибка, выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Функция `checker_main` в этом случае даже не получает управление.

## Переменная `dir_in_path`

```
extern unsigned char *dir_in_path;
```

Данная переменная содержит путь к эталонному рабочему каталогу тестируемой программы. Переменная доступна, только если макрос `USE_TGZ` установлен в ненулевое значение.

## Переменная `dir_out_path`

```
extern unsigned char *dir_out_path;
```

Данная переменная содержит путь к рабочему каталогу тестируемой программы. Переменная доступна, только если макрос `USE_TGZ` установлен в ненулевое значение.

## Переменная `test_info`

```
struct testinfo_struct
{
    int cmd_argc;
    unsigned char **cmd_argv;
    unsigned char *comment;
    unsigned char *team_comment;
};
typedef struct testinfo_struct testinfo_t;
```

```
extern testinfo_t test_info;
```

Переменная `test_info` содержит дополнительную информацию о тесте, заданную в `.inf`-файле. Переменная `test_info` доступна, если макрос `NEED_INFO` установлен в ненулевое значение. Если в проверяющую программу передано неверное количество параметров командной строки, или при открытии файла с дополнительной информацией о тесте возникла ошибка, выполнение проверяющей программы немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Функция `checker_main` в этом случае даже не получает управление.

Поля структуры `testinfo_t` имеют следующее назначение.

- **cmd\_argc.** Содержит количество аргументов командной строки, которые были переданы в тестируемую программу при её запуске.
- **cmd\_argv.** Аргументы командной строки, которые были переданы в тестируемую программу при её запуске.
- **comment.** Дополнительный комментарий к тесту. Текст комментария добавляется в судейский протокол тестирования.
- **team\_comment.** Дополнительный комментарий к тесту. Текст комментария добавляется в пользовательский протокол тестирования.

### 3.2.4 Макросы распределения памяти

```
#define XCALLOC(p,s)
#define XREALLOC(p,s)
#define XALLOCA(p,s)
#define XALLOCAZ(p,s)
#define XMEMMOVE(d,s,c)
#define XMEMZERO(d,c)
```

Данные макросы предоставляют удобный интерфейс к функциям выделения динамической памяти.

#### Макрос `XCALLOC`

Макрос `XCALLOC(p, s)` выделяет динамическую память и обнуляет её. Если попытка выделения динамической памяти завершилась ошибкой, процесс немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. Таким образом можно считать, что выделение памяти с помощью данного макроса всегда завершается успешно. `p` должно быть переменной, элементом массива или полем структуры (то есть `p` — это адресуемое выражение, которое может находиться в левой части оператора присваивания — *lvalue*). `p` должно иметь указательный тип. Выражение для `p` не должно иметь побочных эффектов (то есть не должно использовать операцию присваивания, `++` или `--`). `s` — выражение целочисленного типа, которое не обязано быть *lvalue* или не иметь побочных эффектов. Если `p` имеет тип `TYPE *`, то вызов макроса `XCALLOC(p, n)` выделяет динамическую память под массив из `n` элементов типа `TYPE`. Например,

```
int *p;

XCALLOC(p, 10);
```

выделит память под массив из 10 элементов типа `int` и присвоит адрес начала массива переменной `p`. Выделенная память может быть освобождена явно с помощью вызова функции `free`.

### Макрос **XREALLOC**

Макрос `XREALLOC(p, s)` изменяет размер выделенного блока памяти. Если попытка изменения размера динамической памяти завершилась ошибкой, процесс немедленно завершается с кодом возврата `RUN_CHECK_FAILED`. `p` должно быть `lvalue` указательного типа без побочных эффектов. `s` — это выражение целого типа — новое количество элементов в массиве. Если размер области памяти увеличивается, добавленная память не инициализируется. Выделенная память может быть освобождена явно с помощью вызова функции `free`.

### Макрос **XALLOCA**

Макрос `XALLOCA(p, s)` выделяет память в стековой области. Выделенная память не инициализируется. Если запрашиваемый размер памяти превышает возможности системы, процесс немедленно завершается либо по сигналу `SIGSEGV` (Segmentation fault), либо завершится с кодом возврата `RUN_CHECK_FAILED`. В обоих случаях будет диагностирована внутренняя ошибка тестировщика. `p` должно быть `lvalue` указательного типа без побочных эффектов. `s` — это выражение целого типа — количество элементов в массиве. Поскольку память выделяется на стеке, она будет освобождена автоматически при завершении работы функции, в которой используется `XALLOCA`. Функция `free` использоваться не должна.

### Макрос **XALLOCAZ**

Макрос `XALLOCAZ(p, s)` выделяет память в стековой области и инициализирует её нулями. В остальном данный макрос полностью аналогичен `XALLOCA`.

### Макрос **XMEMMOVE**

Макрос `XMEMMOVE(d, s, c)` копирует массив из `c` элементов, начинающийся с адреса `s`, по адресу `d`. Области памяти могут накладываться произвольным образом. `d` должно быть выражением указательного типа без побочных эффектов. `s` — это выражение указательного типа.

### Макрос **XMEMZERO**

Макрос `XMEMZERO(d, c)` очищает массив из `c` элементов типа `TYPE`, начинающийся с адреса `s`. `s` должно быть выражением указательного типа `TYPE *`.

## 3.2.5 Функции выдачи диагностики

### Функция **checker\_OK**

```
void checker_OK(void);
```



Данная функция печатает на стандартный поток вывода сообщение "OK" и завершает проверяющую программу с кодом возврата 0, сигнализируя программе run о том, что тестируемая программа дала правильный ответ на тесте. Данная функция никогда не возвращает управление в проверяющую программу.

### Функция **fatal**

```
void fatal(int code, char const *format, ...);
```

Данная функция печатает на стандартный поток вывода сообщение в соответствии со спецификацией формата `format` для функции `printf` и дополнительными аргументами, затем печатается символ перехода на новую строку, затем выполнение проверяющей программы завершается с кодом возврата `code`. Данная функция никогда не возвращает управление в проверяющую программу.

### Функция **fatal\_CF**

```
void fatal_CF(char const *format, ...);
```

Данная функция печатает на стандартный поток вывода сообщение в соответствии со спецификацией формата `format` для функции `printf` и дополнительными аргументами, затем печатается символ перехода на новую строку, затем выполнение проверяющей программы завершается с кодом возврата `RUN_CHECK_FAILED`. Данная функция никогда не возвращает управление в проверяющую программу.

Данная функция может использоваться, чтобы диагностировать внутреннюю ошибку тестирования, например, если входные данные теста не удовлетворяют ограничениям задачи.

```
if (N >= 1000) fatal_CF("Value_%d_of_N_is_out_of_range", N);
```

### Функция **fatal\_PE**

```
void fatal_PE(char const *format, ...);
```

Данная функция печатает на стандартный поток вывода сообщение в соответствии со спецификацией формата `format` для функции `printf` и дополнительными аргументами, затем печатается символ перехода на новую строку, затем выполнение проверяющей программы завершается с кодом возврата `RUN_PRESENTATION_ERR`. Данная функция никогда не возвращает управление в проверяющую программу.

Данная функция может использоваться, тогда, когда тестируемая программа не соблюдает формат выходных данных (Presentation error).

```
if (fscanf(f_team, "%d", &N) != 1)
    fatal_PE("Cannot_read_N");
```

### Функция **fatal\_WA**

```
void fatal_WA(char const *format, ...);
```

Данная функция печатает на стандартный поток вывода сообщение в соответствии со спецификацией формата `format` для функции `printf` и дополнительными аргументами, затем печатается символ перехода на новую строку, затем выполнение проверяющей программы завершается с кодом возврата 5 (Wrong answer). Данная функция никогда не возвращает управление в проверяющую программу.

Данная функция может использоваться, тогда, когда тестируемая программа выдаёт неправильный ответ на тест (Wrong answer).

```
if (team_ans != corr_ans)
    fatal_WA("Wrong_answer:_team:_%d,_corr:_%d", team_ans, corr_ans);
```

## 3.2.6 Функции распределения памяти

### Функция `xmalloc`

```
void *xmalloc(size_t size);
```

Данная функция — оболочка над стандартной библиотечной функцией `malloc`. В случае, если `malloc` возвращает `NULL`, проверяющая программа завершается немедленно с кодом возврата `RUN_CHECK_FAILED`. Таким образом, функция `xmalloc` никогда не возвращает `NULL`. Память, выделенная `xmalloc` может освобождаться с помощью `free`, либо перераспределяться с помощью `xrealloc`, `realloc`.

### Функция `xcalloc`

```
void *xcalloc(size_t nmemb, size_t size);
```

Данная функция — оболочка над стандартной библиотечной функцией `calloc`. В случае, если `calloc` возвращает `NULL`, проверяющая программа завершается немедленно с кодом возврата `RUN_CHECK_FAILED`. Таким образом, функция `xcalloc` никогда не возвращает `NULL`. Память, выделенная `xcalloc` может освобождаться с помощью `free`, либо перераспределяться с помощью `xrealloc`, `realloc`.

### Функция `xrealloc`

```
void *xrealloc(void *ptr, size_t size);
```

Данная функция — оболочка над стандартной библиотечной функцией `realloc`. В случае, если `realloc` возвращает `NULL`, проверяющая программа завершается немедленно с кодом возврата `RUN_CHECK_FAILED`. Таким образом, функция `xrealloc` никогда не возвращает `NULL`. Память, выделенная `xrealloc` может освобождаться с помощью `free`, либо перераспределяться с помощью `xrealloc`, `realloc`.

### Функция `xstrdup`

```
unsigned char *xstrdup(const unsigned char *str);
```

Данная функция — оболочка над стандартной библиотечной функцией `strdup`. В случае, если `strdup` возвращает `NULL`, проверяющая программа завершается немедленно с кодом возврата `RUN_CHECK_FAILED`. Таким образом, функция `xstrdup` никогда не возвращает `NULL`. Память, выделенная `xstrdup` может освобождаться с помощью `free`, либо перераспределяться с помощью `xrealloc`, `realloc`.

### 3.2.7 Проверка конца файла

```
void checker_in_eof(void);  
void checker_team_eof(void);  
void checker_corr_eof(void);
```

Эти функции проверяют, достигнут ли конец файла в данном файле. Пропускаются все пробельные символы, если после этого в файле остались непобельные символы, выполнение проверяющей программы завершается с соответствующей ошибкой.

Функция `checker_in_eof` проверяет условие конца файла во входном тестовом файле, если конец файла не достигнут, проверяющая программа завершается с кодом `RUN_CHECK_FAILED`.

Функция `checker_team_eof` проверяет условие конца файла в файле результата тестируемой программы. Если конец файла не достигнут, проверяющая программа завершается с кодом `RUN_PRESENTATION_ERR`.

Функция `checker_corr_eof` проверяет условие конца файла в эталонном файле результата. Если конец файла не достигнут, проверяющая программа завершается с кодом `RUN_CHECK_FAILED`.

### 3.2.8 Заккрытие файла

```
void checker_in_close(void);  
void checker_team_close(void);  
void checker_corr_close(void);
```

Эти функции закрывают соответствующий входной файл. `checker_in_close` закрывает тестовый файл, `checker_team_close` закрывает файл результата тестируемой программы, `checker_corr_close` закрывает эталонный файл результата.

### 3.2.9 Чтение целых и вещественных данных

```
int checker_read_in_int(const unsigned char *, int, int *);  
int checker_read_in_long_long(const unsigned char *, int, long long *);  
int checker_read_in_double(const unsigned char *, int, double *);  
int checker_read_in_long_double(const unsigned char *, int, long double *);  
int checker_read_team_int(const unsigned char *, int, int *);  
int checker_read_team_long_long(const unsigned char *, int, long long *);  
int checker_read_team_double(const unsigned char *, int, double *);  
int checker_read_team_long_double(const unsigned char *, int, long double *);  
int checker_read_corr_int(const unsigned char *, int, int *);  
int checker_read_corr_long_long(const unsigned char *, int, long long *);  
int checker_read_corr_double(const unsigned char *, int, double *);  
int checker_read_corr_long_double(const unsigned char *, int, long double *);
```

Эта группа функций считывает одно целое или вещественное число из одного из входных файлов. Функции имеют следующий общий вид:

```
int checker_read_FILE_TYPE(const unsigned char *name,  
                           int eof_error_flag,  
                           TYPE *p_var);
```

В имени функции `FILE` обозначает файл, из которого считывается число. `in` — входной тестовый файл, `team` — результат работы тестируемой программы, `corr` — эталонный файл результата. `TYPE` — тип считываемого значения.

Параметр `name` содержит имя считываемой переменной. Оно печатается, если чтение значения переменной завершилось ошибкой. Параметр `eof_error_flag` определяет реакцию на достижение конца файла. Если параметр имеет ненулевое значение, и при попытке чтения из входного файла число считано не было, но был достигнут конец файла, выполнение проверяющей программы завершается с ошибкой. Код ошибки зависит от типа входного файла. Если параметр `eof_error_flag` равен 0, при достижении конца файла функция возвращает -1. Параметр `p_var` — это адрес, по которому нужно поместить считанное значение.

Функция возвращает 1, если число было успешно считано, либо -1, если был достигнут конец файла, а параметр `eof_error_flag` равен нулю. Во всех остальных случаях выполнение проверяющей программы завершается с кодом возврата, зависящем от считываемого файла. Если считывается входной тестовый файл `in` или эталонный файл результата `Vcorr`, код возврата равен `RUN_CHECK_FAILED`. Если считывается файл результата работы тестируемой программы, код возврата равен `RUN_PRESENTATION_ERROR`.

```
int checker_read_int(int, const unsigned char *, int, int *);
int checker_read_unsigned_int(int, const unsigned char *, int,
                               unsigned int *);
int checker_read_long_long(int, const unsigned char *, int, long long *);
int checker_read_unsigned_long_long(int, const unsigned char *, int,
                                     unsigned long long *);
int checker_read_double(int, const unsigned char *, int, double *);
int checker_read_long_double(int, const unsigned char *, int, long double *);
```

Эта группа функций считывает одно целое или вещественное число из входного файла, задаваемого параметром функции. Функции имеют следующий общий вид:

```
int checker_read_TYPE(int ind,
                     const unsigned char *name,
                     int eof_error_flag,
                     TYPE *p_var);
```

Здесь параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение ведётся из файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата. В остальном поведение этих функций аналогично поведению функций группы `checker_read_FILE_TYPE`.

### 3.2.10 Чтение текста

#### Функция `checker_skip_eoln`

```
int checker_skip_eoln(int ind, int eof_error_flag);
```

Данная функция пропускает все символы в входном файле до символа перехода на новую строку `'\n'` включительно.

Здесь параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение

ведётся из файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата.

Параметр `eof_error_flag` определяет реакцию, когда конец файла был достигнут ранее, чем символ `'\n'`. Если параметр имеет ненулевое значение, то в такой ситуации выполнение проверяющей программы завершается. Если параметр равен 0, функция возвращает -1.

Функция возвращает 0, если выполнение функции завершилось без ошибок, и -1, если был достигнут конец файла, и параметр `eof_error_flag` равен 0. Во всех остальных случаях выполнение проверяющей программы завершается с кодом возврата, зависящем от считываемого файла. Если считывается входной тестовый файл `in` или эталонный файл результата `Vcorr`, код возврата равен `RUN_CHECK_FAILED`. Если считывается файл результата работы тестируемой программы, код возврата равен `RUN_PRESENTATION_ERROR`.

### Функция `checker_read_file`

```
void checker_read_file(int ind, unsigned char **out, size_t *out_size);
```

Функция считывает файл, задаваемый параметром `ind` целиком в память. Память под считываемый файл выделяется в куче с помощью функции `xrealloc`. Функция не возвращает результат, а при возможной ошибке ввода вывода выполнение проверяющей программы завершается с кодом возврата `RUN_CHECK_FAILED`. Функция корректно считывает произвольные файлы, в том числе файлы, содержащие нулевые байты.

Параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение ведётся из файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата. Параметр `out` — это адрес переменной указательного типа, в которую будет записан адрес начала буфера считанного файла. Параметр `out_size` — адрес переменной, в которую будет записан размер считанного файла.

Под буфер файла выделяется на 1 байт больше, чем размер файла, и в этот дополнительный байт записывается 0. Таким образом, если считываемый файл не содержит в середине нулевых байтов, с буфером считанного файла можно будет работать как с символьной строкой.

### Функция `checker_read_buf`

```
int checker_read_buf(int ind,  
                     const unsigned char *name,  
                     int eof_error_flag,  
                     unsigned char *buf,  
                     size_t buf_size);
```

Данная функция считывает символьную строку в буфер ограниченного размера. При чтении сначала пропускаются все пробельные символы, затем чтение символьной строки ведётся до первого пробельного символа, который остаётся во входном потоке. Если длина символьной строки во входном файле превышает размер буфера, выполнение проверяющей программы завершается с соответствующим кодом возврата.

Параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение ведётся из

файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата.

Параметр `name` позволяет задать имя считываемой переменной. Оно печатается, если выполнение функции завершается из-за ошибки.

Параметр `eof_error_flag` определяет реакцию, когда конец файла был достигнут ранее, чем считан хотя бы один значащий символ строки. Если параметр имеет ненулевое значение, то в такой ситуации выполнение проверяющей программы завершается. Если параметр равен 0, функция возвращает -1.

В параметре `buf` передаётся адрес начала буфера для считывания строки. Если `buf` — нулевой указатель, функция завершает выполнение проверяющей программы с кодом возврата `RUN_CHECK_FAILED`.

В параметре `buf_size` передаётся размер буфера. Если размер буфера равен 0 или превышает 100000, функция завершает выполнение проверяющей программы с кодом возврата `RUN_CHECK_FAILED`. Значение размера буфера на 1 превышает максимальную длину строки, которая может быть записана в этот буфер из-за нулевого байта-терминатора строки. Например, если размер буфера — 80 байтов, то максимальная длина строки, которая может быть в него записана — 79 байтов. Если длина считанной из файла строки превышает максимальную длину, выполнение проверяющей программы завершается с соответствующим кодом возврата.

Функция возвращает длину считанной строки и -1, если был достигнут конец файла, и параметр `eof_error_flag` равен 0. Во всех остальных случаях выполнение проверяющей программы завершается с кодом возврата, зависящем от считываемого файла. Если считывается входной тестовый файл `in` или эталонный файл результата `Vcorr`, код возврата равен `RUN_CHECK_FAILED`. Если считывается файл результата работы тестируемой программы, код возврата равен `RUN_PRESENTATION_ERROR`.

## Функция `checker_read_line`

```
int checker_read_line(int ind,
                      const unsigned char *name,
                      int eof_error_flag,
                      unsigned char **out_str);
```

Данная функция считывает одну текстовую строку из входного файла. Текстовая строка — это последовательность произвольных байтов, завершающихся байтом `'\n'`, который считается входящим в строку. Память под считываемую строку текста выделяется с помощью функции `xrealloc`. В буфер строки после символа `'\n'` добавляется нулевой байт. Если строка не содержит нулевых байтов в середине, то с ней можно работать как с символьной строкой `Si`.

Параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение ведётся из файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата.

Параметр `name` позволяет задать имя считываемой переменной. Оно печатается, если выполнение функции завершается из-за ошибки.

Параметр `eof_error_flag` определяет реакцию, когда конец файла был достигнут ранее, чем считан хотя бы один символ строки текста. Если параметр имеет ненулевое значение, то в такой ситуации выполнение проверяющей программы завершается. Если параметр равен 0, функция возвращает -1.

Параметр `out_str` — это адрес указательной переменной, в которую будет записан адрес начала буфера строки.

Функция возвращает длину считанной строки и -1, если был достигнут конец файла, и параметр `eof_error_flag` равен 0. Во всех остальных случаях выполнение проверяющей программы завершается с кодом возврата, зависящем от считываемого файла. Если считывается входной тестовый файл `in` или эталонный файл результата `Vcorr`, код возврата равен `RUN_CHECK_FAILED`. Если считывается файл результата работы тестируемой программы, код возврата равен `RUN_PRESENTATION_ERROR`.

Длина считанной строки, возвращаемая функцией, корректна, даже если в середине считанной строки присутствуют нулевые байты. В этом случае нельзя полагать, что длина считанной строки равна `strlen(*out_str)`.

### Функция `checker_normalize_line`

```
void checker_normalize_line(unsigned char *str);
```

Данная функция «нормализует» поданную на вход символьную строку `str`. Нормализация заключается в отбрасывании пробельных символов (в том числе `'\n'`) от конца символьной строки. Поскольку длина строки явно не передаётся, символьная строка `str` не должна содержать нулевых байтов в середине строки.

### Функция `checker_read_file_by_line`

```
void checker_read_file_by_line(int ind,
                               unsigned char ***out_lines,
                               size_t *out_lines_num);
```

Данная функция считывает файл целиком в память. При чтении файл разбивается на строки текста, используя `'\n'` как разделитель строк. Считываемый файл не должен содержать нулевые байты.

Параметр `ind` определяет файл, из которого будет считываться значение. Если `ind` равен 0, чтение ведётся из входного тестового файла (`in`). Если `ind` равен 1, чтение ведётся из файла результата тестируемой программы. Если `ind` равен 2, чтение ведётся из эталонного файла результата.

Параметр `out_lines` — это адрес переменной, в которую будет записан адрес начала массива строк. Каждый элемент массива строк — это указатель на символьную строку, считанную из файла.

Параметр `out_lines_num` — это адрес переменной, в которую будет записано количество считанных строк, то есть количество элементов в массиве строк.

Функция не возвращает результата. При возникновении ошибки чтения из файла выполнение проверяющей программы завершается с кодом возврата `RUN_CHECK_FAILED`. Если при чтении файла был считан нулевой байт, выполнение проверяющей программы завершается с соответствующим кодом возврата. Если считывается входной тестовый файл `in` или эталонный файл результата `Vcorr`, код возврата равен `RUN_CHECK_FAILED`. Если считывается файл результата работы тестируемой программы `team`, код возврата равен `RUN_PRESENTATION_ERROR`.

### Функция `checker_normalize_file`

```
void checker_normalize_file(unsigned char **out_lines,
                           size_t *out_lines_num);
```

Данная функция «нормализует» файл, считанный в память с помощью функции `checker_read_file_by_line`. При этом нормализуется каждая строка файла отбрасыванием пробельных символов, находящихся в конце строки. После этого отбрасываются все строки нулевой длины, находящиеся в конце считанного файла. При этом количество строк в считанном файле уменьшается.

Параметр `out_lines` — это адрес начала массива строк, полученный от функции `checker_read_file_by_line`. Параметр `out_lines_num` — это адрес переменной, хранящей количество строк в считанном файле, тоже полученное от функции `checker_read_file_by_line`. В результате работы функции `checker_normalize_file` значение по адресу, указанному в данном параметре, может уменьшиться. Для пустого файла это значение равно 0.

### 3.2.11 Примеры

#### Сравнение двух целых чисел

Приведённая ниже проверяющая программа предполагает, что файл с результатом работы программы должен содержать единственное целое число. Это число сравнивается с эталонным ответом.

```
1  #define NEED_CORR 1
2  #define NEED_INFO 0
3  #define NEED_TGZ 0
4  #include "checker.h"
5
6  int checker_main(int argc, char **argv)
7  {
8      int team_ans, corr_ans;
9
10     // presentation error, если файл пуст или нельзя считать число
11     checker_read_team_int("team_ans", 1, &team_ans);
12
13     // checker failed, если файл пуст или нельзя считать число
14     checker_read_corr_int("corr_ans", 1, &corr_ans);
15
16     // presentation error, если после числа в файле идёт "мусор"
17     checker_team_eof();
18
19     // checker failed, если после числа в файле идёт "мусор"
20     checker_corr_eof();
21
22     if (team_ans != corr_ans)
23         fatal_WA("Answers do not match: team = %d, corr = %d",
24                 team_ans, corr_ans);
25     checker_OK();
26 }
```



## Сравнение двух текстовых файлов

Приведённая ниже проверяющая программа сравнивает файл с результатом работы тестируемой программы с эталонным файлом. При сравнении игнорируются пробелы на концах строк и пустые строки в конце файлов.

```
1  #define NEED_CORR 1
2  #define NEED_INFO 0
3  #define NEED_TGZ 0
4  #include "checker.h"
5
6  int checker_main(int argc, char **argv)
7  {
8      unsigned char **team_lines, **corr_lines;
9      size_t team_lines_num, corr_lines_num, i;
10
11     // считываем файл результата работы программы
12     checker_read_file_by_line(1, &team_lines, &team_lines_num);
13     // считываем эталонный файл
14     checker_read_file_by_line(2, &corr_lines, &corr_lines_num);
15     // отбрасываем пробелы в результате работы программы
16     checker_normalize_file(team_lines, &team_lines_num);
17     // отбрасываем пробелы в эталонном файле
18     checker_normalize_file(corr_lines, &corr_lines_num);
19
20     if (team_lines_num != corr_lines_num)
21         fatal_WA("Different_number_of_lines:_team_=%zu,_corr_=%zu",
22                 team_lines_num, corr_lines_num);
23     for (i = 0; i < team_lines_num; i++)
24         if (strcmp(team_lines[i], corr_lines[i]) != 0)
25             fatal_WA("Line_%zu_differs:_team:\n>%s<\ncorr:\n>%s<",
26                     i + 1, team_lines[i], corr_lines[i]);
27
28     checker_OK();
29 }
```

# Глава 4

## Веб-интерфейс пользователя

В настоящей главе описывается интерфейс пользователя системы **ejudge**, предоставляемый посредством веб-браузера. На стороне сервера веб-интерфейс реализован с помощью CGI-программ **master**, **judge**, **team**, **register** и **users**.

### 4.1 Интерфейс администратора турнира **master**

Веб-интерфейс для администратора турнира предоставляется с помощью CGI-программы **master**.

#### 4.1.1 Конфигурационные файлы

При запуске программа **master** считывает настройки из конфигурационного файла. Каталог, в котором должен находиться данный конфигурационный файл устанавливается при компиляции системы **ejudge**. Настройки по умолчанию таковы, что конфигурационный файл должен находиться в каталоге `../cgi-data` относительно каталога, в котором находится CGI-программа **master**, который определяется по переменной окружения `SCRIPT_FILENAME`. Эта переменная окружения устанавливается веб-сервером **apache** при запуске CGI-программ.

При запуске CGI-программы **master** обязательно должен быть указан идентификатор турнира. Поддерживаются следующие способы указания идентификатора турнира:

- Непосредственно в имени CGI-программы.
- С помощью CGI-переменной `contest_id` указанием её значения в URL или в переменных формы.
- В конфигурационном файле программы.

Идентификатор турнира определяется при запуске CGI-программы **master** по следующим правилам.

1. Если имя CGI-программы имеет вид `master-N`, где  $N$  — положительное целое число, тогда идентификатор турнира полагается равным  $N$ . Возможное указание идентификатора турнира в URL или переменных HTML-формы игнорируется. Далее программа **master** пытается считать конфигурационный файл из каталога конфигурационных файлов. При этом по очереди проверяются следующие возможности:

- (a) Если в каталоге конфигурационных файлов существует файл `serve.cfg`, он будет использоваться в качестве конфигурационного.
- (b) Если предыдущее условие не выполнено, но в каталоге конфигурационных файлов существует файл с именем `master- $N$` , где  $N$  — идентификатор турнира, записанный без ведущих нулей, то такой файл будет использоваться в качестве конфигурационного.
- (c) Если предыдущие условия не выполнены, но в каталоге конфигурационных файлов существует файл с именем `master- $N'$` , где  $N'$  — идентификатор турнира, дополненный до 6 десятичных знаков ведущими незначащими нулями, такой файл будет использоваться в качестве конфигурационного.
- (d) Если предыдущие условия не выполнены, но в каталоге конфигурационных файлов существует файл с именем `master- $N''$` , где  $N''$  — идентификатор турнира, записанный в точности так, как в имени CGI-программы, такой файл будет использоваться в качестве конфигурационного.

Например, если в запросе указано имя CGI-программы `master-05`, то идентификатор турнира полагается равным 5, и в качестве конфигурационного файла выбирается первый файл из списка `master.cfg`, `master-5.cfg`, `master-000005.cfg`, `master-05.cfg`, который существует в каталоге конфигурационных файлов CGI-программы `master`.

Если в конфигурационном файле программы `master` устанавливается идентификатор турнира с помощью (недокументированной) конфигурационной переменной `contest_id`, её значение должно совпадать со значением, заданным в имени программы.

Чтобы использовать задание идентификатора турнира непосредственно в имени CGI-программы, в каталоге CGI-программ должен существовать исполняемый файл `master- $N$` , который может быть копией программы `master`, либо жёсткой или символической ссылкой на неё. Если в каталоге CGI-программ веб-сервера исполняемый файл с именем `master- $N$`  отсутствует, веб-сервер сгенерирует диагностическую страницу несуществующего ресурса.

Обратите внимание, что в конфигурационных файлах сервера `apache` можно указывать правила переписывания URL (URL Rewriting Rules), с помощью которых можно выполнять сколь угодно сложные трансформации URL. В этом случае обратитесь, пожалуйста, к документации на веб-сервер `apache`.

2. Если имя CGI-программы равно в точности `master`, тогда идентификатор турнира берётся из CGI-переменных, переданных в URL или в переменных HTML-формы. Например, идентификатор турнира может быть задан непосредственно в URL следующим образом: `http://host/cgi-bin/master?contest_id=5`. Пусть идентификатор турнира равен  $N$ . Тогда для поиска конфигурационного файла программы `master` проверяются следующие возможности.
  - (a) Если в каталоге конфигурационных файлов существует файл с именем `master- $N$` , где  $N$  — идентификатор турнира, записанный без ведущих нулей, то такой файл будет использоваться в качестве конфигурационного.

- (b) Если предыдущее условие не выполнено, но в каталоге конфигурационных файлов существует файл с именем `master- $N'$` , где  $N'$  — идентификатор турнира, дополненный до 6 десятичных знаков ведущими незначащими нулями, такой файл будет использоваться в качестве конфигурационного.
- (c) Если предыдущие условия не выполнены, но в каталоге конфигурационных файлов существует файл `serve.cfg`, он будет использоваться в качестве конфигурационного.

Например, если URL запроса имеет вид `http://host/cgi-bin/master?contest_id=5`, то идентификатор турнира полагается равным 5, и в качестве конфигурационного файла выбирается первый файл из списка `master-5.cfg`, `master-000005.cfg`, `master.cfg`, который существует в каталоге конфигурационных файлов CGI-программы **master**.

Если в конфигурационном файле программы **master** устанавливается идентификатор турнира с помощью (недокументированной) конфигурационной переменной `contest_id`, её значение должно совпадать со значением, заданным в имени программы.

3. Если имя CGI-программы **master** имеет любую другую форму, отличную от описанных выше, то в каталоге конфигурационных файлов должен существовать конфигурационный файл `M.cfg` в котором должна быть определена конфигурационная переменная `contest_id`. Здесь  $M$  — это имя, под которым запускается CGI-программа **master**. Возможное указание идентификатора турнира в URL или переменных HTML-формы игнорируется.

Например, если URL запроса имеет вид `http://host/cgi-bin/master-mycontest`, то в каталоге конфигурационных файлов должен существовать файл `master-mycontest.cfg`, который будет использован в качестве конфигурационного файла CGI-программы **master**.

Обратите внимание, что выбор конфигурационного файла происходит до его чтения и анализа его содержимого. Таким образом, если в процессе поиска конфигурационного файла был найден некоторый конфигурационный файл, который, однако, оказался некорректным, то программа **master** сгенерирует диагностическую страницу неверного конфигурационного файла, даже если в списке возможных конфигурационных файлов далее находился корректный конфигурационный файл.

Заметим, что такие правила определения идентификатора турнира используются всеми CGI-программами, а не только программой **master**.

#### 4.1.2 Вход в систему

Для входа в систему в строке URL браузера введите URL `http://HOST/cgi-bin/master`, где *HOST* — IP-адрес или имя компьютера, на котором установлена система **ejudge**. Если система **ejudge** настроена правильно, будет выдано приглашение ко вводу регистрационного имени, пароля и идентификатора турнира. Экран браузера со страницей приглашения показан на рис. 4.1. В поле «Login» необходимо ввести регистрационное имя, в поле «Password» — пароль (он не отображается на экране) и в поле «Contest ID» — идентификатор турнира. Когда все поля заполнены необходимо нажать на кнопку «Submit».

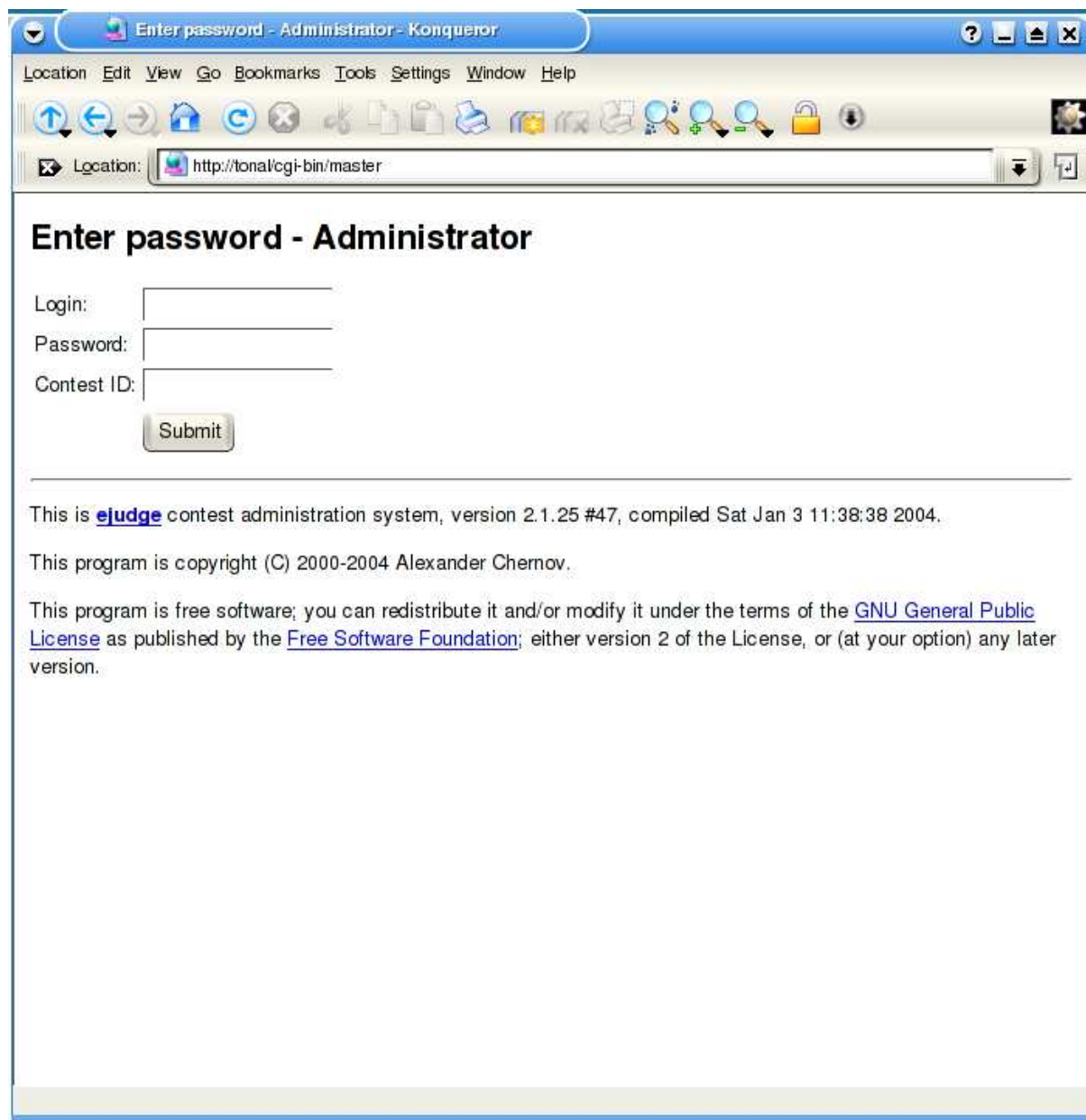


Рис. 4.1: Страница входа в систему (турнир неизвестен)

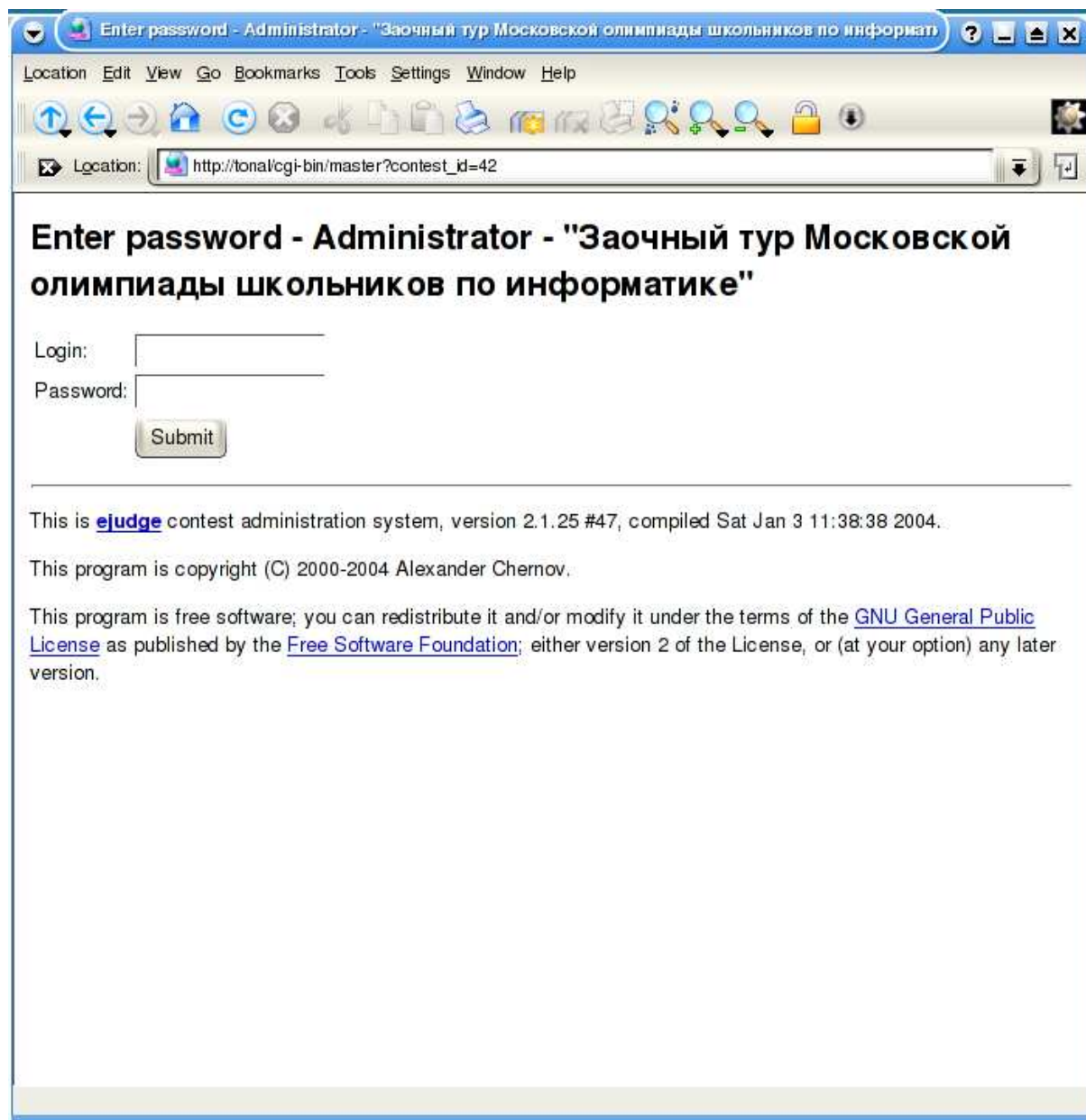


Рис. 4.2: Страница входа в систему (турнир задан)

Введённые данные проверяются на корректность, и в случае их правильности происходит переход к главному экрану программы **master**. Если введённые данные некорректны, будет отображена диагностическая страница ошибки. Более точно причину ошибки можно определить в диагностике, печатаемой программами **userlist-server**, **serve**, либо в диагностике веб-сервера. Возможные причины ошибки входа в систему перечислены ниже:

- Введено некорректное регистрационное имя пользователя (такой пользователь не существует в базе пользователей).
- Введён неправильный пароль.
- Указан недопустимый идентификатор турнира. Идентификатор турнира является положительным целым числом. Возможно, что такого турнира в системе не существует.
- Пользователь с указанным регистрационным именем не имеет полномочий использовать CGI-программу **master**. Полномочия устанавливаются в элементе **cap** конфигурационного файла турнира **contest.xml**. За возможность использования программы **master** отвечает бит полномочий **MASTER\_LOGIN**.
- Пользователь с указанным регистрационным именем не зарегистрирован как участник указанного турнира. Несмотря на то, что этот пользователь использует привилегированную программу **master**, он всё равно должен быть зарегистрирован на турнир как и рядовой участник.
- Программа **master** для данного турнира не может быть использована с IP-адреса клиента.
- Не существует или неверен конфигурационный файл программы **serve**.
- Ошибка в конфигурационном файле турнира.
- Не запущена программа управления турниром **serve**.

Возможно задание идентификатора турнира непосредственно в URL. В этом случае URL имеет вид `http://HOST/cgi-bin/master?contest_id=N`, где *HOST* — IP-адрес или имя компьютера, на котором установлена система **ejudge**, а *N* — идентификатор турнира. В этом случае приглашение ко вводу регистрационного имени и пароля имеет вид, показанный на рис. 4.2. Обратите внимание, что отсутствует поле ввода идентификатора турнира, кроме того в заголовке страницы печатается название турнира. В поле «Login» необходимо ввести регистрационное имя, в поле «Password» — пароль (он не отображается на экране). Когда все поля заполнены необходимо нажать на кнопку «Submit».

Если с IP-адреса клиента запрещено использование программы **master** для указанного турнира *N*, то диагностическая страница будет выдана до приглашения ко вводу регистрационного имени. Другие возможные причины ошибки входа в систему перечислены выше.

### 4.1.3 Управление турниром до его начала

На рис. 4.3 показаны элементы управления турниром, доступные до начала турнира. Они выделены красным подчёркиванием.

1. Состояние турнира “The contest is not started” — турнир не начат.



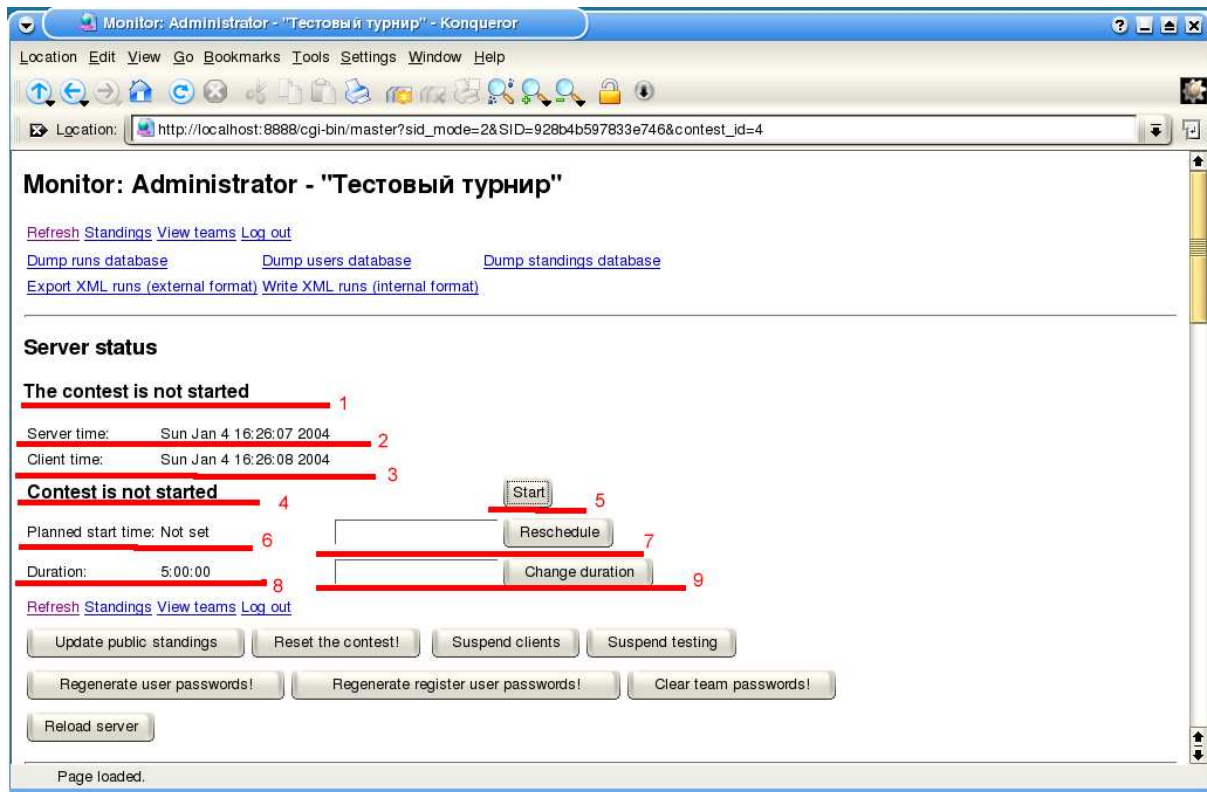


Рис. 4.3: Элементы управления турниром (турнир не начался)

2. Астрономическое время программы-сервера турнира **serve**. Время считывается из файла состояния сервера, который обновляется каждую секунду. Таким образом можно определить, что программа **serve** обслуживает данный турнир.

- Если турнир обслуживается программой **serve**, работающей самостоятельно, и время сервера слишком отличается от времени клиента (см. ниже), CGI-программа **master** диагностирует ошибку "Server is down" и генерирует соответствующую диагностическую страницу.
- Если программа **serve** работает под управлением программы супер-сервера **super-serve**, которая запускает программу-сервер некоторого турнира по требованию, то есть когда поступает запрос к этому турниру, время клиента и время сервера могут при первом обращении сильно отличаться. В этом случае время сервера показывает, когда последний раз обновлялся файл состояния турнира. Ошибка "Server is down" в этом случае не диагностируется.

Обратите внимание, что для работы CGI-программ с турниром под управлением **super-serve** в конфигурационном файле турнира **contest.xml** в элементе **client\_flags** необходимо установить флаг клиентов **IGNORE\_TIME\_SKEW**, например, следующим образом.

```
<client_flags>IGNORE_TIME_SKEW</client_flags>
```

3. Астрономическое время в CGI-программе **master**, которая в данный момент обрабатывает запрос на генерацию html-страницы, поступивший от веб-браузера. К сожалению, "Client time" — неудачное название, так как это время не имеет никакого отношения к



времени на компьютере пользователя, на котором запущен веб-браузер, демонстрирующий данную страницу.

4. Состояние турнира — турнир не начат.
5. Кнопка начала турнира. С помощью этой кнопки можно в любой момент времени начать турнир.

Чтобы пользователь мог стартовать турнир, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

6. Здесь отображается планируемое время начала турнира. Оно может быть не установлено (отображается “Not set” как на рисунке), либо может быть установлено на любое время в пределах текущих суток. В момент, когда астрономическое время становится не меньше, чем время, указанное в данной клетке таблицы, турнир начинается автоматически.
7. С помощью данного элемента управления можно установить время автоматического начала турнира. Для этого нужно ввести время в поле ввода и затем нажать на кнопку “Reschedule”. Время начала турнира вводится в формате  $H[:M]$ , где  $H$  — часы, а  $M$  — минуты. Часы вводятся в 24-часовом формате, то есть представляют собой целое число от 0 до 23. Минуты — это целое число от 0 до 59. Минутная часть вместе с двоеточием может быть опущена. Если устанавливается время суток, меньшее текущего, турнир стартует автоматически немедленно.

Чтобы пользователь мог устанавливать время автоматического начала турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

8. В данной ячейке таблицы отображается установленная продолжительность турнира. Продолжительность турнира отображается либо как “Unlimited”, если продолжительность турнира не ограничена, либо в формате  $HH:MM:SS$ , где  $HH$  — количество полных часов,  $MM$  — количество полных минут в неполном последнем часе,  $SS$  — количество секунд в последней минуте.
9. Данный элемент управления позволяет изменить установленную продолжительность турнира. В поле ввода необходимо ввести новую продолжительность турнира, затем нажать кнопку “Change duration”. Продолжительность турнира задаётся в виде  $H[:M]$ , где  $H$  — количество часов, а  $M$  — количество минут. Количество минут должно находиться в интервале от 0 до 59. Минутная составляющая вместе с двоеточием может быть опущена. Максимальная продолжительность турнира, которую можно установить с помощью данного элемента управления, равна 1000000 минутам, что соответствует примерно 694 суткам. Продолжительность турнира неограниченной продолжительности не может быть изменена. Продолжительность турнира ограниченной продолжительности не может быть сделана неограниченной.

Начальная продолжительность турнира устанавливается в конфигурационном файле сервера турнира `serve.cfg` с помощью глобальной конфигурационной переменной `contest_time`.

Чтобы пользователь мог изменять продолжительность турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

Остальные элементы управления могут использоваться, когда турнир уже стартовал, и рассматриваются далее.

#### 4.1.4 Управление турниром в течение турнира

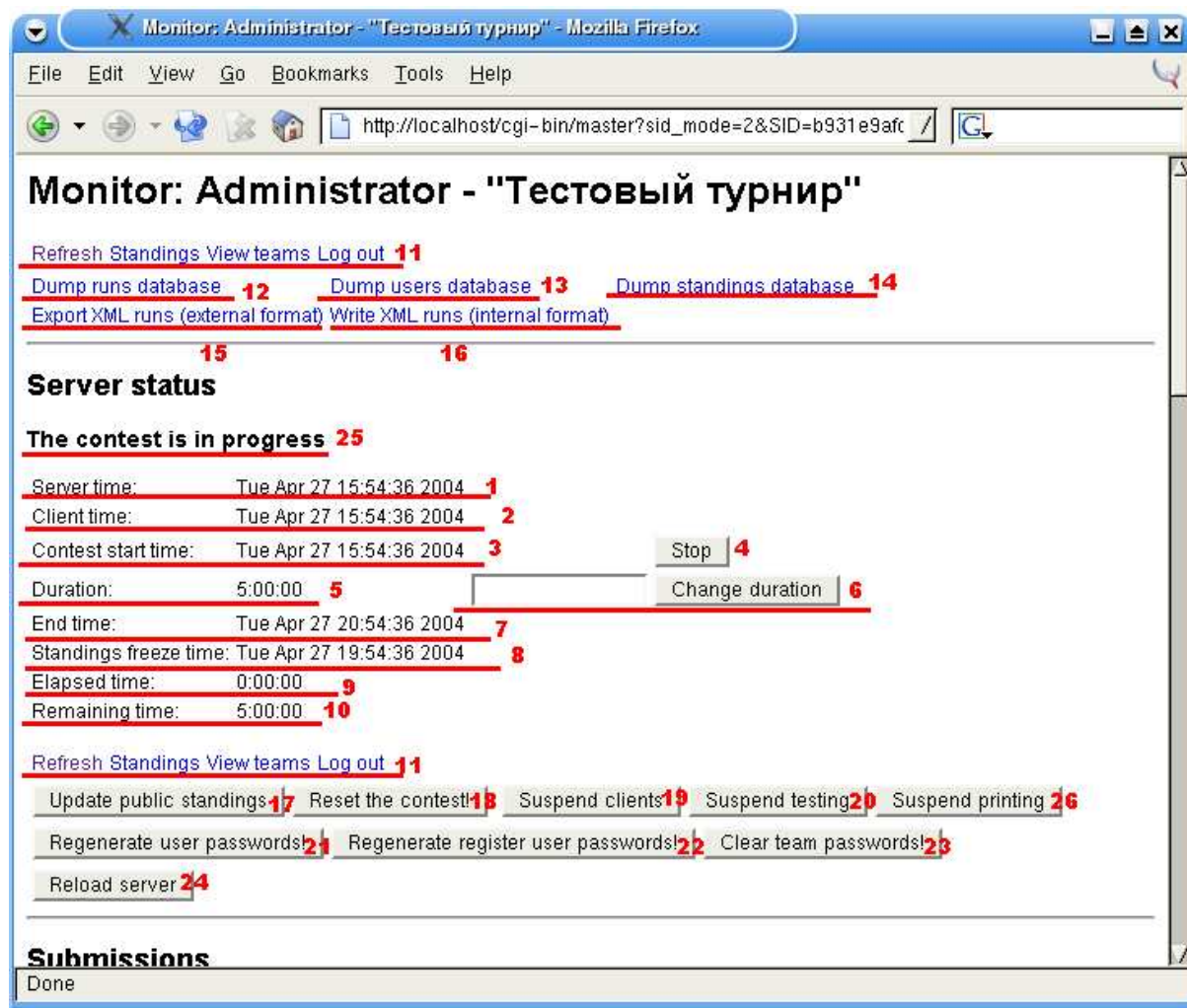


Рис. 4.4: Элементы управления турниром

На рис. 4.4 показаны элементы управления турниром, доступные в ходе турнира. Все выделенные на рисунке элементы управления описываются ниже.

1. Астрономическое время программы-сервера управления турниром `serve`. Полное описание элемента “Server time” приведено [выше](#).
2. Астрономическое время в CGI-программе `master`, которая в данный момент обрабатывает запрос на генерацию html-страницы, поступивший от веб-браузера. См. также полное описание [выше](#).

3. Астрономическое время начала турнира.
4. С помощью кнопки “Stop” можно завершить турнир в любой момент до истечения времени турнира.

Чтобы пользователь мог останавливать турнир, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

5. Установленная продолжительность турнира отображается либо как “Unlimited”, если продолжительность турнира не ограничена, либо в формате *HH:MM:SS*, где *HH* — количество полных часов, *MM* — количество полных минут в неполном последнем часе, *SS* — количество секунд в неполной последней минуте. Когда текущая продолжительность турнира превышает установленную продолжительность турнира, турнир завершается автоматически.
6. Данный элемент управления позволяет изменить установленную продолжительность турнира. В поле ввода необходимо ввести новую продолжительность турнира, затем нажать кнопку “Change duration”. Продолжительность турнира задаётся в виде *H[:M]*, где *H* — количество часов, а *M* — количество минут. Количество минут должно находиться в интервале от 0 до 59. Минутная составляющая вместе с двоеточием может быть опущена. Максимальная продолжительность турнира, которую можно установить с помощью данного элемента управления, равна 1000000 минутам, что соответствует примерно 694 суткам. Продолжительность турнира неограниченной продолжительности не может быть изменена. Продолжительность турнира ограниченной продолжительности не может быть сделана неограниченной.

Продолжительность турнира конечной продолжительности может быть как увеличена, так и уменьшена. При этом на уменьшение продолжительности накладывается такое ограничение, что новая продолжительность турнира не должна быть такой, что турнир должен был бы закончиться в некоторый момент в прошлом. Другими словами, уменьшение продолжительности турнира не должно приводить к его немедленному завершению.

Начальная продолжительность турнира устанавливается в конфигурационном файле сервера турнира `serve.cfg` с помощью глобальной конфигурационной переменной `contest_time`.

Чтобы пользователь мог изменять продолжительность турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

7. Здесь отображается ожидаемое время завершения турнира, которое вычисляется по времени начала турнира (3) и текущей продолжительности турнира (5). В этот момент времени турнир будет завершён автоматически, если только до этого момента он не будет завершён явно с помощью кнопки “Stop” (4). Изменения в текущей продолжительности турнира с помощью кнопки “Change duration” (7) немедленно отражаются в ожидаемом времени завершения турнира.

Для турниров неограниченной продолжительности данная информация не отображается.

8. Здесь отображается ожидаемое время «заморозки» таблицы текущих результатов турнира. От момента заморозки и до окончания турнира таблица текущих результатов автоматически не обновляется при поступлении новых решений. Время в минутах от момента заморозки до конца турнира задаётся с помощью глобальной конфигурационной переменной `stand_freeze_time` (переменная имеет альтернативное имя `board_fog_time`) в конфигурационном файле сервера турнира `serve.cfg`. Если значение этой переменной установлено в 0, заморозка таблицы результатов отключается. В этом случае, а также в случае турниров неограниченной продолжительности данная информация не отображается.

Изменения в текущей продолжительности турнира с помощью кнопки “Change duration” (7) немедленно отражаются в ожидаемом времени заморозки таблицы результатов. В любой момент времени таблица текущих результатов турнира может быть обновлена вручную с помощью кнопки “Update public standings” (17).

Автоматическое обновление результатов турнира может быть полностью отключено установкой глобальной конфигурационной переменной `autoupdate_standings` в значение 0 в файле конфигурации турнира `serve.cfg`.

9. Здесь отображается время турнира, прошедшее от его начала.
10. Здесь отображается время, оставшееся до конца турнира.
11. Ссылки быстрой навигации по страницам управления турниром.
- Нажатие на ссылку “Refresh” приведёт к обновлению информации на главной странице управления турниром.
  - Нажатие на ссылку “Standings” вызовет переход на страницу отображения текущих результатов турнира. Текущие результаты турнира, отображаемые на этой странице, никогда не замораживаются. Обычные участники турнира, а также наблюдатели не имеют доступа к данной «привилегированной» таблице результатов. Чтобы пользователь программы `master` имел возможность смотреть привилегированные текущие результаты турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `VIEW_STANDINGS`.
  - Нажатие на ссылку “View teams” вызовет переход на страницу отображения информации об участниках турнира. Данная страница подробно рассматривается ниже. Чтобы пользователь имел полномочия просматривать список участников турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `LIST_CONTEST_USERS`.
  - Нажатие на ссылку “Log out” приведёт к завершению сеанса работы администратора турнира. При этом идентификатор сессии (SID), отображаемый в URL станет недействительным. Для входа на страницы управления турниром необходимо вновь ввести регистрационное имя и пароль на странице входа в программу.
12. Нажатие на ссылку “Dump runs database” вызовет печать базы посылок в формате CSV (comma-separated values). В формате CSV каждая запись размещается на отдельной текстовой строке, а поля записи разделяются символом ‘;’. Данные в формате CSV

могут быть импортированы, например, в реляционные базы данных или в программы электронных таблиц для дальнейшей обработки.

База посылок в формате CSV отображается непосредственно в окне браузера (для неё устанавливается тип содержимого (Content-type) “text/plain”). База посылок может быть сохранена в текстовом файле средствами браузера. Поля файла базы посылок в формате CVS описываются в соответствующем разделе.

Данная операция требует специальных привилегий. Для того, чтобы пользователь мог получить базу посылок в формате CSV, для пользователя в файле конфигурации турнира `contest.xml` в элементе `cap` должен быть установлен бит `DUMP_RUNS`.

13. Нажатие на ссылку “Dump users database” вызовет печать базы участников турнира в формате CSV. База участников турнира в формате CSV отображается непосредственно в окне браузера (для неё устанавливается тип содержимого (Content-type) “text/plain”). База участников турнира может быть сохранена в текстовом файле средствами браузера. Поля файла базы участников турнира в формате CVS описываются в соответствующем разделе.

Данная операция требует специальных привилегий. Для того, чтобы пользователь мог получить базу участников турнира в формате CSV, для пользователя в файле конфигурации турнира `contest.xml` в элементе `cap` должен быть установлен бит `DUMP_USERS`.

14. Нажатие на ссылку “Dump standings database” вызовет печать таблицы текущих результатов турнира в формате CSV. Таблица текущих результатов турнира в формате CSV отображается непосредственно в окне браузера (для неё устанавливается тип содержимого (Content-type) “text/plain”). Таблица текущих результатов турнира может быть сохранена в текстовом файле средствами браузера. Поля таблицы текущих результатов турнира в формате CVS описываются в соответствующем разделе.

Данная операция требует специальных привилегий. Для того, чтобы пользователь мог получить базу участников турнира в формате CSV, для пользователя в файле конфигурации турнира `contest.xml` в элементе `cap` должен быть установлен бит `DUMP_STANDINGS`.

15. Нажатие на ссылку “Export XML runs (external format)” вызовет генерацию XML-файла с базой посылок. XML-файл отображается непосредственно в окне браузера (для него устанавливается тип содержимого (Content-type) “text/plain”) и затем может быть сохранён в файле средствами браузера.

Экспортируемый XML-формат базы посылок отличается от внутреннего (см. ниже) тем, что в экспортируемый формат добавляется информация о зарегистрированных участниках турнира, сконфигурированных языках программирования и задачах турнира. По такому XML-файлу можно однозначно воспроизвести ход турнира и его результаты. С другой стороны, в экспортируемом формате отсутствует внутренняя информация о посылке, такая как IP-адрес, размер, хэш-код и пр. Экспортируемый формат предназначен для отчёта о ходе турнира.

Данная операция требует специальных привилегий. Для того, чтобы пользователь мог получить базу посылок в формате CSV, для пользователя в файле конфигурации турнира `contest.xml` в элементе `cap` должен быть установлен бит `DUMP_RUNS`.



16. Нажатие на ссылку “Write XML runs (internal format)” вызовет генерацию XML-файла с базой посылок во внутреннем формате. XML-файл отображается непосредственно в окне браузера (для него устанавливается тип содержимого (Content-type) “text/plain”) и затем может быть сохранён в файле средствами браузера.

Внутренний формат отличается от внешнего (см. выше) тем, что во внутреннем формате для каждой посылки содержится полная информация о посылке, включающая IP-адрес, контрольную сумму, размер, внутренние флаги и т. д. Внутренний формат не содержит информации о зарегистрированных участниках, языках программирования и задачах. Внутренний формат предназначен для синхронизации базы посылок (и как следствие таблицы текущих результатов) в случаях, когда один и тот же турнир проводится в нескольких точках проведения с отдельным сервером турнира в каждой точке проведения. В этом случае предполагается, что база зарегистрированных участников турнира, список языков программирования и задач во всех точках проведения одинаковы.

База посылок во внутреннем формате может быть проимпортирована с помощью элемента управления “Import and merge XML runs log” (см. ниже).

Данная операция требует специальных привилегий. Для того, чтобы пользователь мог получить базу посылок в формате CSV, для пользователя в файле конфигурации турнира `contest.xml` в элементе `cap` должен быть установлен бит `DUMP_RUNS`.

17. Кнопка “Update public standings” предназначена для ручного обновления текущих результатов турнира, доступных для участников турнира и прочих зрителей из сети Интернет. С помощью неё можно обновлять результаты даже тогда, когда автоматическое обновление отключено в период заморозки результатов. Поскольку операция обновления результатов необратима, после нажатия на данную кнопку будет выдана страница подтверждения операции. Для выполнения обновления результатов на странице подтверждения необходимо нажать на кнопку “Yes, update standings”.

Чтобы пользователь мог вручную обновлять текущее положение команд, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

18. Кнопка “Reset the contest” полностью сбрасывает турнир. Полностью очищается база посылок и база сообщений, удаляются все файлы в архиве сообщений и архиве посылок, продолжительность турнира сбрасывается на значение, установленное в глобальной конфигурационной переменной `contest_time` конфигурационного файла турнира `serve.cfg`.

Поскольку операция сброса турнира необратима, после нажатия на данную кнопку будет выдана страница подтверждения операции. Для выполнения обновления результатов на странице подтверждения необходимо нажать на кнопку “Yes, reset the contest”.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

19. Кнопка “Suspend clients” приостанавливает обработку сервером турнира `serve` запросов от CGI-программы `team`, запросы от привилегированных программ `master` и `judge` продолжают обрабатываться. В режиме приостановленной обработки участникам турнира будет генерироваться html-страница, показывающая только состояние турнира.

Участники турнира не могут в это время просматривать список своих посылок, список сообщений судьям и от судей, сами сообщения. Участники не могут посылать решения на проверку и посылать сообщения судьям.

В режиме приостановленной обработки запросов участников в разделе состояния турнира (25) выводится дополнительная строка “Team requests are suspended”, доступная и участника турнира. В режиме приостановленной обработки кнопка “Suspend clients” заменяется на кнопку “Resume clients”, нажав на которую можно выключить режим приостановленной обработки запросов участников.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

20. Кнопка “Suspend testing” приостанавливает автоматическую проверку поступающих решений. В режиме приостановленной проверки решений участники турнира могут выполнять все обычные действия, включая и посылку решений на проверку, но посланное на проверку решение не будет автоматически отправлено на компиляцию и последующее тестирование. Вместо этого оно получит статус “Available for testing”, в котором будет находиться до тех пор, пока не будет пересужено каким-либо способом. Данная возможность полезна, когда требуется уменьшить нагрузку на сервер турнира (особенно, если сервер турнира одновременно выполняет компиляцию и тестирование решений) в последние минуты турнира, когда поток решений особенно интенсивен.

Посылку со статусом “Available for testing” можно пересудить одним из следующих способов:

- Выбрав опцию “Rejudge” в строке состояния данной посылки на главной странице программы **master**.
- Выбрав опцию “Rejudge” в режиме просмотра посылки (“View source”), доступном из главной страницы программы **master**.
- Нажав на кнопку “Rejudge all” (см. ниже). При этом будут пересужены все решения, находящиеся в базе посылок.
- Выбрав соответствующую задачу и нажав на кнопку “Rejudge!” в разделе “Rejudge problem” (см. ниже). В этом случае будут пересужены все решения указанной задачи.
- Нажав на кнопку “Judge suspended runs” (см. ниже). В этом случае будут пересужены все решения в базе посылок, которые имеют статус “Available for testing”.

В режиме приостановленного тестирования решений участников в разделе состояния турнира (25) выводится дополнительная строка “Testing of team’s submits is suspended”, доступная и участника турнира. В этом режиме кнопка “Suspend testing” заменяется на кнопку “Resume testing”, нажав на которую можно выключить режим приостановленного тестирования решений участников.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

21. Кнопка “Regenerate user passwords” генерирует случайные пароли участников турнира для всех непривилегированных участников. Пароли привилегированных участников остаются без изменений.

В системе **ejudge** каждый пользователь может иметь два независимых пароля: регистрационный пароль и пароль участника турнира. Регистрационный пароль генерируется автоматически при регистрации пользователя в системе **ejudge** и отправляется электронной почтой на адрес, указанный при регистрации. Пользователь может изменить этот пароль с помощью CGI-программы **register**. Регистрационный пароль используется для входа в программу **register**, а также привилегированные программы **master** и **judge**, если пользователь имеет соответствующие привилегии. Регистрационный пароль используется для входа в программу **team**, если не установлен пароль участника.

Пароль участника турнира устанавливается администратором турнира. Пароль участника турнира действует для входа в программу **team**. В текущей версии системы **ejudge** пароль участника турнира един для всех турниров. Если задан и пароль участника, и регистрационный пароль, для входа в программу **team** необходимо использовать пароль участника.

По нажатию на кнопку “Regenerate user passwords” генерируются случайные пароли. Поскольку операция генерации пароля необратима, после нажатия на данную кнопку будет выдана страница подтверждения операции. Для выполнения обновления результатов на странице подтверждения необходимо нажать на кнопку “Yes, generate passwords!”. После генерации паролей страница со всеми сгенерированными паролями отображается веб-браузером. При этом сбрасываются все идентификаторы сессий, созданные для участников турнира. Таким образом, всем участникам турнира придётся заново ввести своё регистрационное имя и пароль.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `GENERATE_TEAM_PASSWORDS`.

22. Кнопка “Regenerate register user passwords” генерирует случайные регистрационные пароли для всех непривилегированных участников турнира. Пароли привилегированных участников остаются без изменений. Поскольку операция генерации паролей необратима, после нажатия на данную кнопку будет выдана страница подтверждения операции. Для выполнения обновления результатов на странице подтверждения необходимо нажать на кнопку “Yes, generate passwords!”. После генерации паролей страница со всеми сгенерированными паролями отображается веб-браузером. При этом сбрасываются все идентификаторы сессий, созданные для участников турнира. Таким образом, всем участникам турнира придётся заново ввести своё регистрационное имя и пароль.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `GENERATE_TEAM_PASSWORDS`.

23. Кнопка “Clear team passwords!” сбрасывает пароли участников турнира для всех непривилегированных участников турнира. Поскольку операция сброса паролей необратима, после нажатия на данную кнопку будет выдана страница подтверждения операции. Для выполнения обновления результатов на странице подтверждения необходимо нажать на кнопку “Yes, clear passwords!”. После этого пароль участника турнира считается не



установленным, и для входа в программу **team** может использоваться регистрационный пароль.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `GENERATE_TEAM_PASSWORDS`.

24. Кнопка “Reload server” действует только тогда, когда сервер турнира **serve** работает под управлением программы **super-serve**. Тогда нажатие на эту кнопку приводит к немедленному завершению работы программ **serve** и **run**, обслуживающих данных турнир. Они будут немедленно перезапущены, и в результате сервер турнира будет перезагружен и конфигурационные файлы считаны заново.

В режиме самостоятельной работы программы **serve** на нажатие этой кнопки будет выдано сообщение об ошибке.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

25. Здесь выводится текущее состояние сервера. Если в текущий момент времени отображение текущих результатов турнира заморожено, выводится сообщение “Standings are frozen”. Кроме того дополнительно выводятся сообщения о приостановке обслуживания клиентов и о приостановке проверки решений.
26. Кнопка “Suspend printing” позволяет приостановить обслуживание запросов на печать, поступающих от непривилегированных участников соревнования. Запросы на печать привилегированных пользователей обслуживаются как обычно. Кнопка отображается, только если глобальная конфигурационная переменная `enable_printing` конфигурационного файла `serve.cfg` установлена в значение `true`.

После нажатия на данную кнопку сервер турнира переходит в режим приостановки обслуживания запросов на печать. В этом режиме все запросы на печать, поступающие от непривилегированных клиентов, немедленно отвергаются. Кроме этого, в CGI-программе **team** перестаёт отображаться ссылка “Print”, позволяющая участникам печатать исходный текст посылок. Кнопка “Suspend printing” заменяется на кнопку “Resume printing” нажатие на которую выключает режим приостановки обслуживания запросов на печать.

#### 4.1.5 Специальные элементы управления турниром

На рис. 4.5 выделены элементы управления турниром, доступные в турнире, проходящем по системе *OLYMPIAD*. Дополнительно, данный турнир имеет неограниченную продолжительность.

1. Данная строка отображает режим проверки решений участников. Турнир системы *OLYMPIAD* может находиться в двух режимах: режиме приёма решений от участников и в режиме проверки решений. В режиме приёма решений от участников поступившее или пересуживаемое решение запускается на приёмочных тестах. Это — первые несколько тестов из полного набора тестов для данной задачи. Результаты запуска решения на приёмочных тестах становятся немедленно доступными участнику.

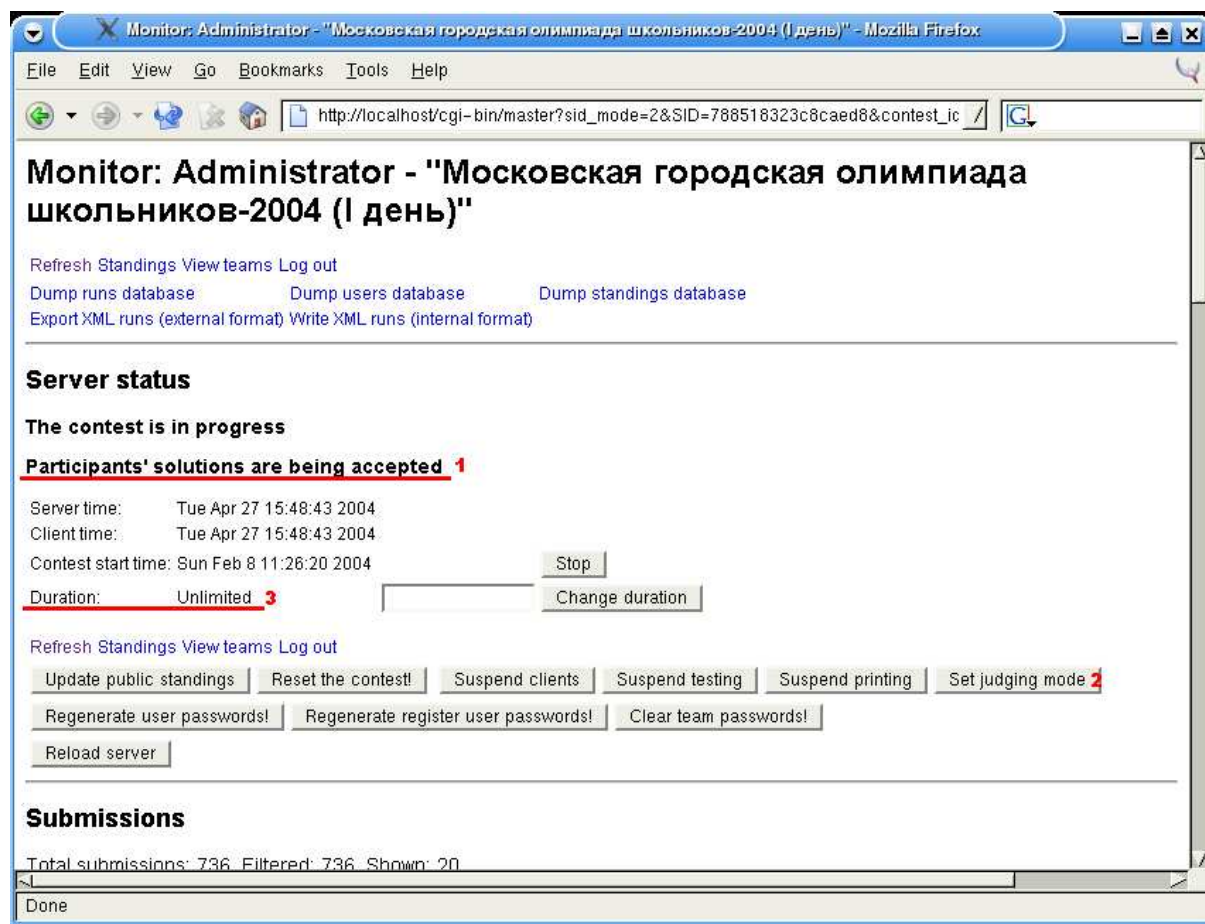


Рис. 4.5: Элементы управления турниром в режиме *OLYMPIAD*

Цель проверки на приёмочных тестах: убедиться, что программа соблюдает форматы входных и выходных данных. В режиме проверки решений ведётся полная проверка решений участников.

Сообщение “Participant’s solutions are being accepted” означает, что сервер турнира находится в режиме приёма решений, а сообщение “Participant’s solutions are being judged” означает, что сервер турнира находится в режиме проверки решений.

2. Кнопка “Set judging mode” позволяет переключить сервер турнира из режима приёма решений в режим проверки решений. В режиме проверки решений данная кнопка трансформируется в кнопку “Set accepting mode”, которая переключает сервер турнира в режим приёма решений.
3. Так отображается продолжительность турнира в случае, когда она не ограничена. Для турниров неограниченной продолжительности кнопка изменения продолжительности “Change duration” не работает.

#### 4.1.6 Управление турниром после его окончания

На рис. 4.6 выделены элементы управления турнира, доступные после окончания турнира. Эти элементы управления описываются ниже.

1. Текущее состояние турнира — турнир завершён.
2. Время начала турнира.
3. Нажатие на кнопку “Continue” вызывает продолжение турнира. Турнир может быть продолжен, если в конфигурационном файле турнира `serve.cfg` глобальная конфигурационная переменная `enable_continue` установлена в `true`. Кроме того, установленная продолжительность турнира должна быть такова, что продолжение турнира не приведёт к немедленному его завершению из-за истечения времени.

Если турнир заканчивается из-за истечения времени, его можно продолжить в два шага: во-первых, увеличить продолжительность турнира, чтобы текущий момент времени был бы до момента завершения турнира, и, во-вторых, продолжить турнир с помощью нажатия на кнопку “Continue”.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

4. Текущая продолжительность турнира.
5. С помощью этого элемента управления можно изменить продолжительность турнира. Изменение продолжительности турнира после его окончания возможно только, если в конфигурационном файле турнира `serve.cfg` глобальная конфигурационная переменная `enable_continue` установлена в `true`.

Для выполнения этой операции пользователем, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `CONTROL_CONTEST`.

6. Фактическое время окончания турнира.

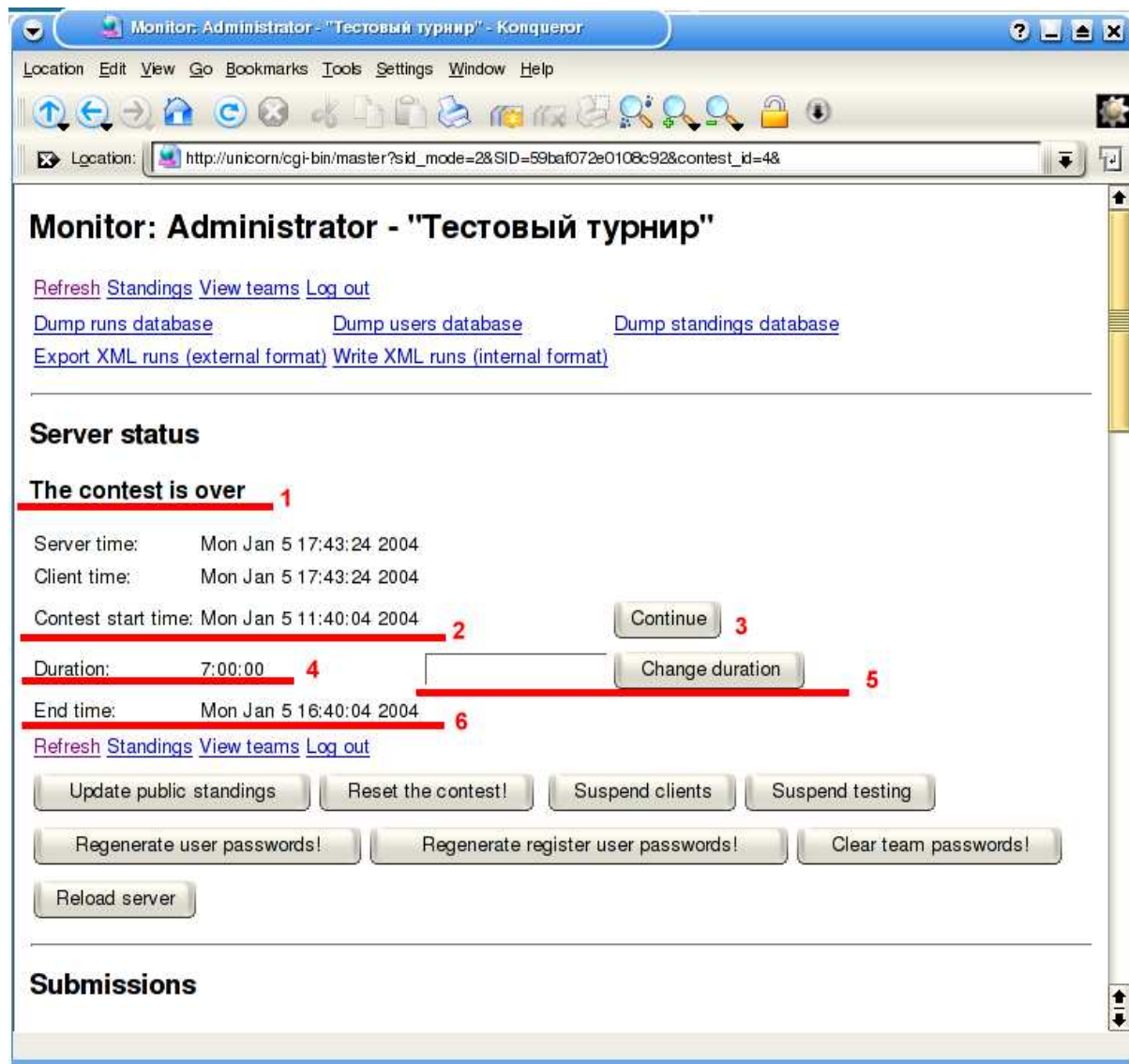


Рис. 4.6: Элементы управления турниром после его окончания

## 4.1.7 Элементы отображения журнала посылок

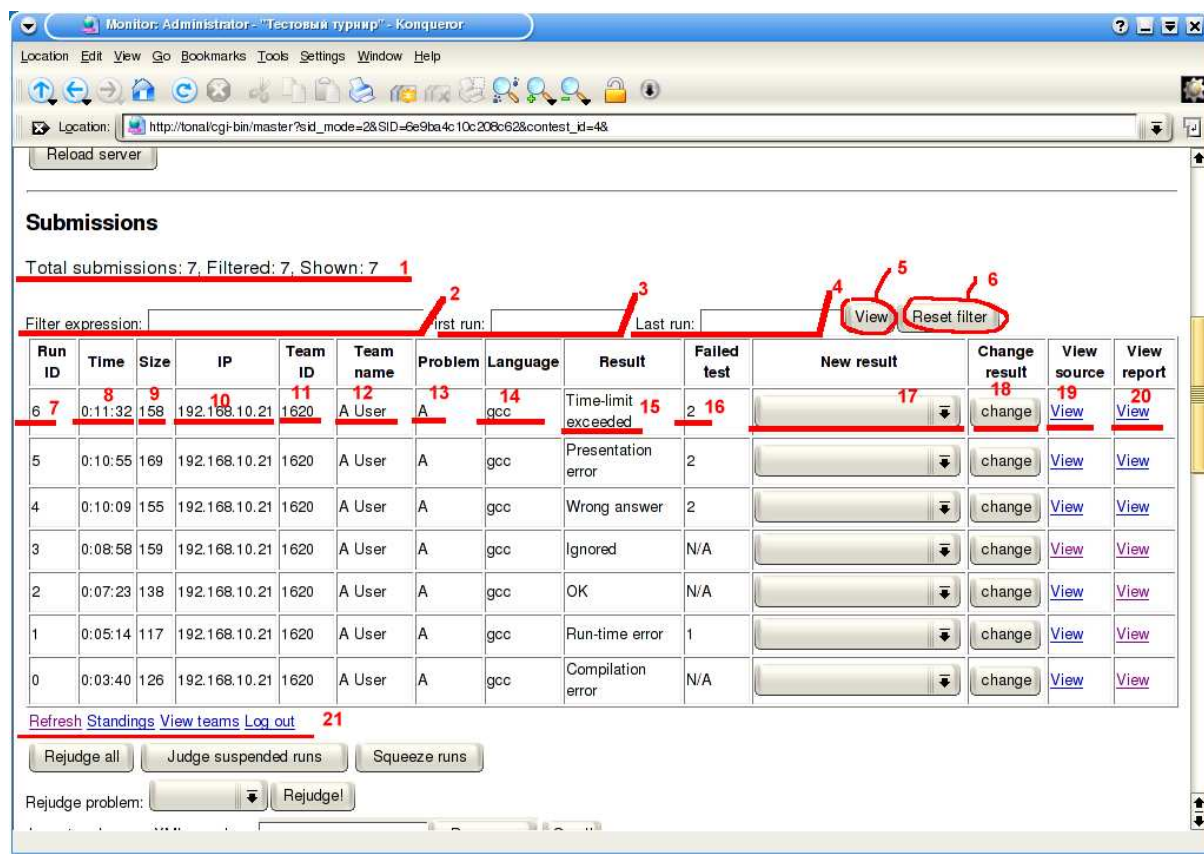


Рис. 4.7: Элементы просмотра и редактирования журнала посылок

На рис. 4.7 приведена часть главной страницы, генерируемой CGI-программой **master**, на которой находятся элементы просмотра и управления журналом посылок. Эти элементы управления доступны как в течение турнира, так и после его завершения. Все элементы подробно описаны ниже.

1. Здесь показывается текущая статистика по посылкам текущего турнира. “Total submissions” — это общее количество посылок в базе посылок. “Filtered” — количество посылок после фильтрации с помощью [выражения фильтра посылок](#). “Shown” — количество отображённых посылок. Если в выражении фильтра посылок не указано количество отображаемых посылок, отображается 20 последних посылок.
2. В этом поле ввода задаётся [выражение фильтра посылок](#). Чтобы выражение фильтра посылок было применено, необходимо нажать кнопку “View” (5). Чтобы сбросить выражение фильтра, необходимо нажать на кнопку “Reset filter” (6). Если выражение фильтра не определено, фильтр посылок отключён.
3. Это поле ввода позволяет задать посылку, которая будет отображаться первой в списке посылок.
4. Это поле ввода позволяет задать посылку, которая будет отображаться последней в списке посылок. Данное поле ввода совместно с полем ввода (3) позволяет задать диапазон отображаемых посылок. Пусть значение поля “First run” равно A, а значение



поля “Last run” равно  $B$ . Выбор диапазона просмотра производится после фильтрации посылок в соответствии с установленным фильтром посылок. Пусть после применения фильтра посылок осталось  $N$  посылок, удовлетворяющих этому фильтру. Посылки нумеруются от 0 и до  $N - 1$ .

Если значение  $A$  или  $B$  отрицательно, то соответствующее значение преобразуется в порядковый номер посылки среди отобранных посылок по формуле  $N + A$ . Таким образом, номер посылки  $-1$  обозначает последнюю посылку среди отобранных.

Пусть  $A'$ ,  $B'$  — номера посылок после описанного выше преобразования. Тогда если  $A' > B'$ , то посылки выводятся в обратном порядке от посылки с большим номером к посылке с меньшим номером, то есть от более «молодой» к более «старой». Более старая посылка будет размещаться внизу страницы. Если  $A' \leq B'$ , то посылки выводятся в прямом порядке от посылки с меньшим номером к посылке с большим номером, то есть от более «старой» к более «молодой». Более новая посылка будет размещаться внизу страницы.

5. К помощью кнопки “View” включается фильтр посылок (2) и выбор диапазона отображаемых посылок (3) и (4).
6. С помощью кнопки “Reset filter” текущие установки фильтра посылок и диапазона отображаемых посылок сбрасываются в значения по умолчанию. По умолчанию фильтр посылок отключён, то есть отображаются все посылки. Отображается не более 20 последних посылок в обратном порядке (от более «молодой» к более «старой»).

Следующие элементы отображения используются для показа информации о посылках.

7. В столбце “Run ID” отображается порядковый номер посылки в базе посылок. Первая посылка имеет номер 0.
8. В столбце “Time” отображается время от начала турнира до момента получения этой посылки. Для турниров неограниченной продолжительности предпочтительнее показывать не время, прошедшее от начала турнира, а астрономическое время. Для этого необходимо установить в значение *true* значение глобальной конфигурационной переменной `show_astr_time` конфигурационного файла турнира `serve.cfg`.
9. В столбце “Size” показывается размер программы в данной посылки в байтах.
10. В столбце “IP” показывается IP-адрес компьютера, с которого была послана посылка.
11. В столбце “Team ID” отображается идентификатор участника турнира. Идентификатор участника турнира — это целое число, большее 0.
12. В столбце “Team name” отображается имя участника турнира. Имя участника вводится самим участником турнира при его регистрации на турнир с помощью CGI-программы `register`. Если участник турнира не установил имя участника, используется регистрационное имя (login) участника.
13. В столбце “Problem” отображается короткое имя задачи, решаемой данной посылкой. Короткое имя задачи берётся из значения конфигурационной переменной `short_name` раздела описания задачи конфигурационного файла `serve.cfg`. Если по тем или иным причинам идентификатору задачи, хранящемуся в базе данных, не соответствует никакой задачи, печатается числовой идентификатор задачи.

14. В столбце “Language” отображается короткое имя языка программирования, использованного в данной посылке. Короткое имя языка программирования берётся из значения конфигурационной переменной `short_name` раздела описания языка программирования конфигурационного файла `serve.cfg`. Если по тем или иным причинам идентификатору языка программирования, хранящемуся в базе данных, не соответствует никакого языка, печатается числовой идентификатор языка программирования.
15. В столбце “Result” отображается статус данной посылки. Возможное множество значений статуса посылки зависит от типов турнира. Все возможные значения статуса перечислены в таблице 5.1.
16. В столбце “Failed test” показывается минимальный номер теста, на котором тестируемая программа дала неверный результат. В случае решения, прошедшего все тесты, либо решения, получившего статус “Compilation error”, “Ignored”, “Accepted for testing”, в данном столбце выводится N/A. Такое правило действует только для турниров в режиме *АСМ*.  
  
Для турниров в режиме *KIROV* и *OLYMPIAD* этот столбец называется “Passed tests” и показывает количество тестов, на которых программа дала правильный ответ. В случаях, когда это неприменимо (см. выше), выводится N/A. В указанных режимах турнира в таблице посылок добавляется новый столбец “Score”, в котором отображается количество баллов, полученное за данную попытку. Количество баллов отображается в формате  $S_1(A) = S_2$ , где  $S_1$  — количество баллов заработанное на тестах,  $A$  — номер попытки,  $S_2$  — количество баллов с учётом штрафа за попытку.
17. В столбце “New result” выводится список возможных статусов посылки, которые можно установить с помощью кнопки “change” (18). Список возможных статусов зависит от режима турнира.
18. С помощью кнопки “change” можно установить статус посылки, выбранный с помощью меню “New result” (17).
19. Ссылка “View source” позволяет переключиться в режим просмотра исходного текста посылки и полной служебной информации об этой посылке. Также в режиме просмотра исходного текста посылки возможно редактирование служебной информации.
20. Ссылка “View report” позволяет переключиться в режим просмотра протокола тестирования посылки.
21. Ссылки быстрой навигации подробно описаны [выше](#).

#### 4.1.8 Элементы управления журналом посылок

На рис. 4.8 показаны элементы управления журналом посылок, доступные администратору турнира. Все элементы управления описываются далее.

1. Ссылки быстрой навигации подробно описаны [выше](#).
2. Команда “Rejudge all” вызывает перетестирование всех посылок, которые могут быть перетестированы. Посылка может быть перетестирована при выполнении следующих условий:

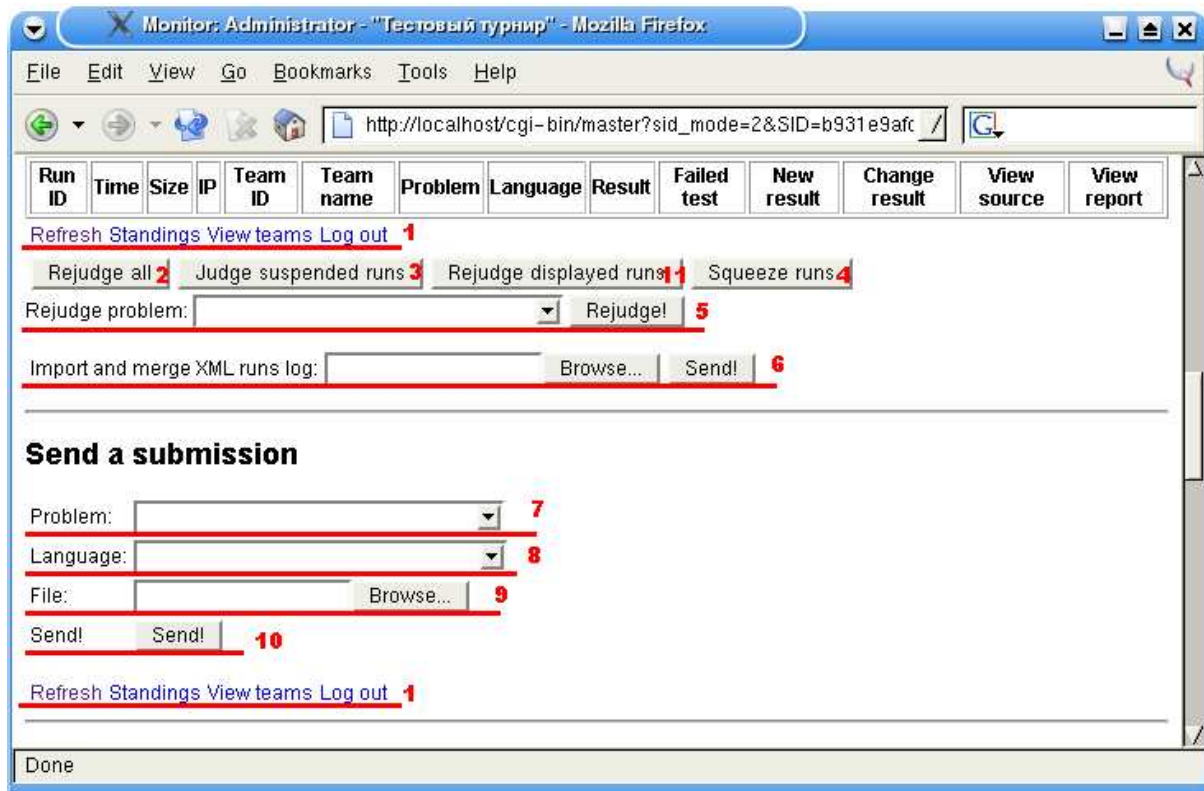


Рис. 4.8: Элементы управления журналом посылок

- поле `is_readonly` посылки установлено в значение *false*;
- поле `is_imported` посылки установлено в значение *false*;
- посылка выполнена по задаче, для которой конфигурационная переменная `disable_testing` секции описания задачи конфигурационного файла турнира `serve.cfg` установлена в значение *false* (значение по умолчанию).

После нажатия на кнопку “Rejudge all” будет выдана страница подтверждения операции. Для действительного выполнения операции на странице подтверждения необходимо нажать кнопку “Yes, rejudge all”.

Команда пересуживания всех посылок доступна только привилегированным пользователям, у которых в данном турнире установлен бит полномочий `EDIT_RUN`.

3. Команда “Judge suspended runs” отправляет на тестирование все посылки, которые могут быть перетестированы (см. выше) и которые в базе посылок имеют статус `ACCEPTED`. Команда может применяться для тестирования всех посылок, которые были накоплены в период приостановленного тестирования, включённый командой “Suspend testing” администратора турнира.

После нажатия на кнопку “Judge suspended runs” будет сгенерирована страница подтверждения операции. Для действительного выполнения операции на странице подтверждения необходимо нажать кнопку “Yes, judge”.

Данная команда доступна только привилегированным пользователям, у которых в данном турнире установлен бит полномочий `EDIT_RUN`.



4. Команда “Squeeze runs” выполняет «чистку» базы посылок. Из неё удаляются все записи со статусом `EMPTY`. При этом у оставшихся посылок может измениться их номер, что требует изменения пути, по которому сохраняются исходный код программы и протоколы тестирования в соответствующих архивных каталогах. Если в базе посылок нет записей со статусом `EMPTY`, состояние базы посылок не изменяется.

После нажатия на кнопку “Squeeze runs” запрашивается подтверждение выполнения данной операции. Чтобы операция была выполнена на странице подтверждения необходимо нажать на кнопку “Yes, squeeze runs”.

Данная команда доступна только привилегированным пользователям, у которых в данном турнире установлен бит полномочий `CONTROL_CONTEST`.

5. Команда “Rejudge problem” позволяет пересудить посылки по выбранной задаче. Пересуживаются только пересуживаемые посылки (см. выше) по пересуживаемым задачам. Необходимо в меню выбрать требуемую задачу, затем нажать на кнопку “Rejudge”. Обратите внимание, что подтверждение операции не запрашивается.

Данная команда доступна только привилегированным пользователям, у которых в данном турнире установлен бит полномочий `EDIT_RUN`.

6. Команда “Import and Merge XML runlog” позволяет проимпортировать в идущий турнир базу посылок во внутреннем XML-формате. Для включения возможности импортирования базы посылок глобальная конфигурационная переменная `enable_runlog_merge` конфигурационного файла турнира `serve.cfg` должна быть установлена в значение `true`.

Данная команда доступна только привилегированным пользователям, у которых в данном турнире установлен бит полномочий `IMPORT_XML_RUNS`.

База посылок во внутреннем XML-формате генерируется с помощью команды “Write XML runs (internal format)” администратора турнира. База посылок во внутреннем формате не содержит информации об участниках, языках программирования и задачах, а только их идентификаторы в записях о посылках, поэтому задачи, языки программирования и участники должны быть согласованы во всех точках проведения турнира, обменивающихся информацией о посылках. Если при импорте будет обнаружено несогласование, импорт не будет выполнен.

При импорте база посылок будет автоматически упорядочена в порядке возрастания астрономического времени приёма посылки и (для одинакового времени) в порядке возрастания идентификатора участника турнира. При импорте учитывается значение поля `is_imported` и флага `authoritative` импортируемой записи. Если в импортируемой базе посылок и локальной базе посылок обнаруживается запись, у которой совпадают время приёма посылки `timestamp`, идентификатор участника `team`, идентификатор языка программирования `language` и идентификатор задачи `problem`, конфликт разрешается следующим образом:

- Если флаг `is_imported` не установлен, и флаг `authoritative` импортируемой посылки также не установлен, локальная посылка имеет приоритет (сохраняется в результирующем журнале посылок), импортируемая посылка игнорируется.
- Если флаг `is_imported` установлен, и флаг `authoritative` не установлен, локальная посылка имеет приоритет, а в случае несовпадения какого-либо из

оставшихся полей локальной и импортируемой посылок выдаётся диагностическое сообщение.

- Если флаг `is_imported` не установлен, и флаг `authoritative` импортируемой посылки установлен, локальная посылка имеет приоритет. Такая ситуация возможна, когда импортируется собственный журнал посылок данного сервера турнира. При несовпадении какого либо из оставшихся полей локальной и импортируемой посылки выдаётся диагностическое сообщение.
- Если флаг `is_imported` установлен, и флаг `authoritative` импортируемой посылки также установлен, импортируемая посылка имеет приоритет.

7. Элементы управления 7–10 предназначены для отправки программ на проверку в привилегированном режиме. Чтобы иметь возможность отправлять программы на проверку в привилегированном режиме у пользователя должен быть установлен бит `SUBMIT_RUN` полномочий в файле конфигурации турнира `contest.xml`.

У привилегированных посылок автоматически устанавливается флаг `is_hidden`, то есть они не участвуют в подсчёте результата и не отображаются в таблице результатов. С другой стороны, привилегированные посылки могут выполняться до начала турнира и после его завершения. Изменить флаг `is_hidden` посылки можно при редактировании данных посылки с помощью интерфейса администратора CGI-программы `master` (см. ниже).

С помощью элемента “Problem” необходимо выбрать задачу для сдачи. Если задача вариантная (то есть значение конфигурационной переменной `variant_num` конфигурационного файла `serve.cfg` больше нуля), привилегированный пользователь может выбрать любой вариант задачи. Крайний срок сдачи задачи (см. конфигурационную переменную задачи `deadline`) игнорируется, то есть для сдачи доступны даже задачи, крайний срок сдачи которых истёк.

Элементы меню выбора имеют вид “ $S - L$ ”, где  $S$  — это короткое имя задачи, определяемое конфигурационной переменной `short_name` раздела описания задачи, а  $L$  — это длинное имя, определяемое конфигурационной переменной `long_name`. Для вариантных задач элементы меню выбора имеют вид “ $S-V - L$ ”, где  $V$  — номер варианта (целое число от 1 до значения конфигурационной переменной `variant_num`).

8. С помощью элемента ввода “Language” необходимо выбрать язык программирования. Привилегированному пользователю доступны все языки программирования, включая языки, у которых установлена конфигурационная переменная `disabled` секции описания языка программирования конфигурационного файла задачи `serve.cfg`.

Элементы меню выбора имеют вид “ $S - L$ ”, где  $S$  — это короткое языка программирования, определяемое конфигурационной переменной `short_name` раздела описания языка программирования, а  $L$  — это длинное имя, определяемое конфигурационной переменной `long_name`.

9. С помощью элемента “File” необходимо выбрать файл с исходным текстом программы. Для этого нужно нажать на кнопку “Browse”, и с помощью стандартного диалога выбора файла выбрать файл с исходным текстом программы.
10. При нажатии на кнопку “Send” программа будет передана серверу турнира, который занесёт её в базу посылок и начнёт её обработку.

11. Кнопка “Rejudge displayed runs” отправляет на перетестирование все послышки, которые отображаются в данном окне в соответствии с выражением фильтра. Все остальные команды пересуживания посылок являются, по сути, частными случаями данной команды.

#### 4.1.9 Элементы просмотра журнала сообщений

На рис. 4.9 представлены элементы интерфейса пользователя для просмотра сообщений, поступающих от участников и ответов судей участникам. Все интерфейсные элементы подробно описаны далее.

Если глобальная конфигурационная переменная `disable_team_clars` установлена в `true`, участники турнира не будут иметь возможность направлять сообщения судьям. Если глобальная конфигурационная переменная `disable_clars` установлена в `true`, то и участники турнира не будут иметь возможность адресовать сообщения судьям, и, наоборот, судьи и администратор турнира не будут иметь возможность писать сообщения участникам. В последнем случае элементы просмотра журнала сообщений и ввода текста нового сообщения не отображаются на страницах администратора турнира и судей турнира.

1. Ссылки быстрой навигации подробно описаны [выше](#).
2. В этой строке показывается общее количество сообщений в базе сообщений (как от участников судьям так и наоборот) и количество отображённых сообщений. По умолчанию отображается 15 последних сообщений, но изменить количество отображаемых сообщений можно с помощью элементов интерфейса 3, 4 и 5.
3. В данном поле ввода указывается номер первого сообщения для просмотра.
4. В данном поле указывается номер последнего сообщения для просмотра.
5. При нажатии на кнопку “View” отображаются сообщения в соответствии со значениями, введёнными в элементах 3 и 4. Пусть в настоящий момент в базе сообщений находятся  $N$  сообщений. Сообщения нумеруются от 0 (самое раннее сообщение) до  $N - 1$ . Пусть в поле “First clar” (3) введено значение  $F$ , а в поле “Last clar” (4) введено значение  $L$ . Сообщения будут отображены в соответствии со следующими правилами.
  - Если значение  $F < 0$ , то это число задаёт номер сообщения, считая от последнего, то есть  $F' = N + F$ , а в противном случае  $F' = F$ .
  - Если теперь  $F' < 0$ , то  $F'' = 0$ , а если  $F' \geq N$ , то  $F'' = N - 1$ , а в противном случае  $F'' = F'$  (проверяется выход за границы допустимых изменений номера сообщения).
  - Аналогичные вычисления проводятся для  $L$ . Если  $L < 0$ , то  $L' = N + L$ , а иначе  $L' = L$ .
  - Если  $L' < 0$ , то  $L'' = 0$ , а если  $L' \geq N$ , то  $L'' = N - 1$ , а иначе  $L'' = L'$ .
  - Если  $F'' \leq L''$ , то будет отображён список сообщений от сообщения с номером  $F''$  до сообщения с номером  $L''$  в порядке их поступления, то есть в порядке увеличения календарного времени их получения сервером турнира. Если  $F'' > L''$ , то отображается список сообщений от сообщения с номером  $L''$  до сообщения с номером  $F''$  в обратном порядке, то есть от сообщений, поступивших позже, к сообщениями, поступившим раньше.

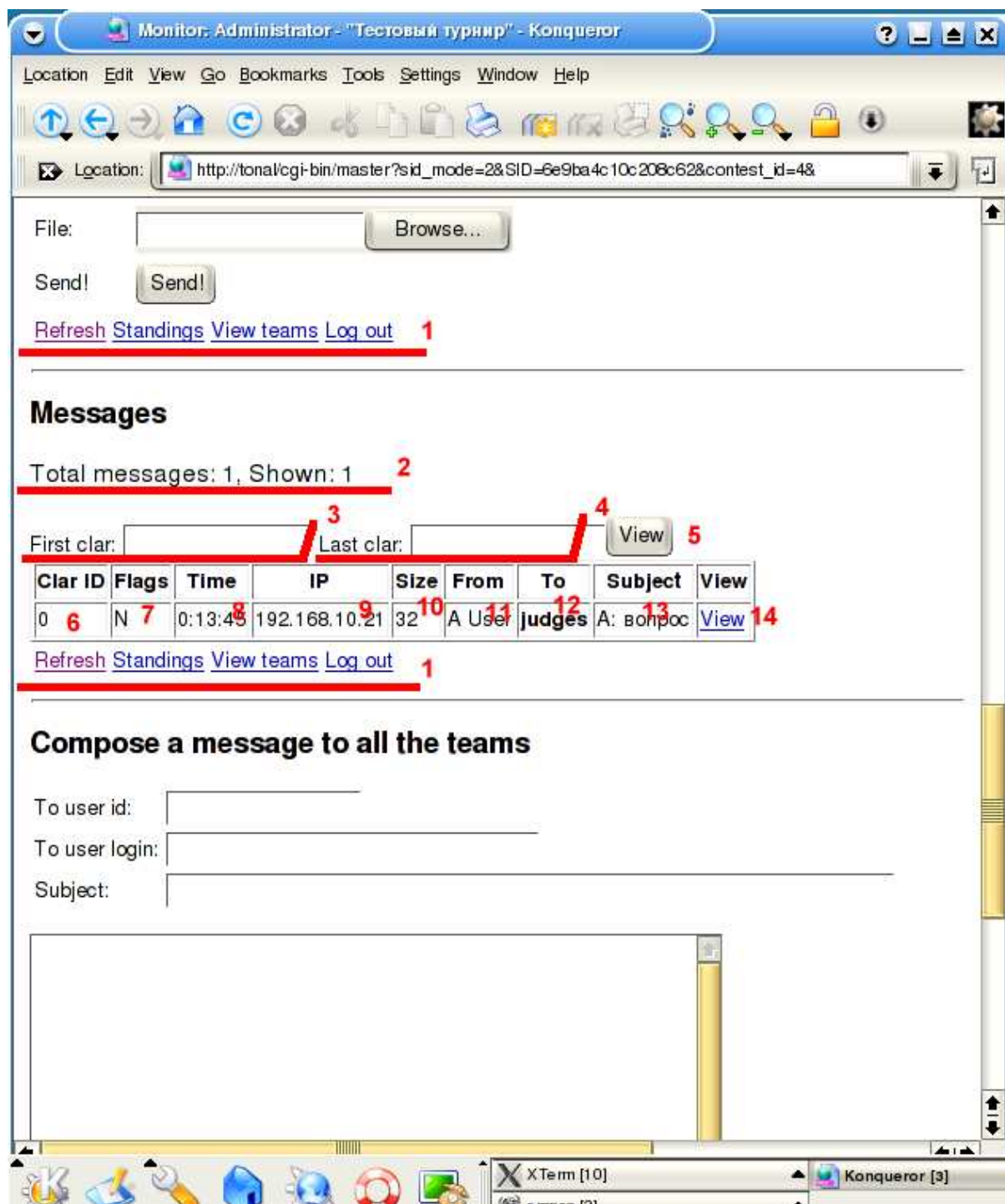


Рис. 4.9: Элементы просмотра журнала сообщений

Таким образом, чтобы отобразить весь список сообщений, начиная с последнего и до самого первого, в поле “First clar” достаточно ввести значение  $-1$ , а в поле “Last clar” — значение  $0$ .

6. Элементы интерфейса 6—14 соответствуют столбцам таблицы сообщений. В столбце “Clar ID” отображается порядковый номер сообщения, который изменяется, как сказано выше, от  $0$  до  $N - 1$ , где  $N$  — общее число сообщений в базе.
7. В столбце “Flags” отображаются флаг состояния сообщения. Флаг состояния может иметь следующие значения:
  - **N.** «Новое» сообщение, то есть сообщение, которое ещё не было просмотрено судьёй турнира. Судьёй в данном контексте считается пользователь, использующий для просмотра сообщений CGI-программу **judge**. Флаг сбрасывается, когда текст сообщения открывается с помощью нажатия на ссылку “View” (14) в CGI-программе **judge**. Просмотры сообщения в CGI-программе **master** не сбрасывают этот флаг.
  - **A.** «Отвеченное» сообщение, то есть сообщение, которое было просмотрено судьёй или администратором турнира с помощью нажатия на ссылку “View” (14), и на которое был написан и отослан автору сообщения или всем участникам турнира ответ. Флаг A устанавливается при использовании как CGI-программы **judge**, так и CGI-программы **master**.

Если в столбце статуса не отображается ни один из описанных выше статусов сообщения, значит данное сообщение не является ни новым, ни отвеченным, то есть оно уже было просмотрено, но ответ на него ещё не был разослан.

8. В столбце “Time” отображается время, прошедшее от начала турнира до момента получения сервером турнира **serve** этого сообщения. Если в файле конфигурации турнира **serve.cfg** установлена глобальная конфигурационная переменная **show\_astr\_time**, в столбце “Time” отображается астрономическое время получения сообщения.
9. В столбце “IP” отображается IP-адрес отправителя сообщения, то есть IP-адрес компьютера, на котором работает браузер, в котором отображаются страницы, генерируемые CGI-программой **master**, **judge** или **team**.
10. В столбце “Size” отображается размер текста сообщения в байтах.
11. В столбце “From” указывается отправитель сообщения. Отправитель может быть участником турнира, использующим интерфейс турнира, поддерживаемый CGI-программой **team**, и в этом случае здесь отображается имя участника турнира. Если отправителем является привилегированный пользователь, использующий CGI-программы **judge** или **master**, в столбце “From” отображается строка **Judges** вне зависимости от конкретного привилегированного пользователя, отославшего данное сообщение.
12. В столбце “To” указывается адресат сообщения. Если сообщение послано судьёй или администратором турнира и адресовано участнику турнира, то в данном поле отображается имя участника турнира. Если сообщение послано судьёй или администратором турнира и адресовано всем участникам турнира, в данном поле отображается строка



All. Если сообщение послано участником турнира судьям, то в данном поле отображается строка Judges. Обычный участник турнира не может послать сообщение всем участникам турнира или какому-либо конкретному участнику.

13. В столбце “Subject” указывается тема сообщения. В таблице сообщений длина строки темы сообщения ограничена 16 символами. Полностью тему сообщения можно посмотреть при просмотре всего сообщения с помощью ссылки “View” (14).
14. С помощью ссылки “View” можно просмотреть полную тему и текст сообщения и написать на него ответ.

#### 4.1.10 Элементы интерфейса посылки сообщений

На рис. 4.10 показаны элементы интерфейса CGI-программы **master**, предназначенные для составления и рассылки сообщений конкретным участникам турнира или всем участникам турнира. Все элементы интерфейса описаны далее.

Если глобальная конфигурационная переменная `disable_team_clars` установлена в *true*, участники турнира не будут иметь возможность направлять сообщения судьям. Если глобальная конфигурационная переменная `disable_clars` установлена в *true*, то и участники турнира не будут иметь возможность адресовать сообщения судьям, и, наоборот, судьи и администратор турнира не будут иметь возможность писать сообщения участникам. В последнем случае элементы просмотра журнала сообщений и ввода текста нового сообщения не отображаются на страницах администратора турнира и судей турнира.

1. В поле “To userid” вводится числовой идентификатор пользователя, которому адресовано сообщение. Если сообщение адресовано всем участникам турнира, в данном поле должно быть введено число 0.
2. В поле “To user login” вводится регистрационное имя (login) пользователя, которому адресовано сообщение. Если сообщение адресовано всем участникам турнира, в данном поле должно быть введена строка `all`.

Поля ввода “To userid” и “To user login” являются взаимоисключающими. Если указан числовой идентификатор участника турнира, то указание регистрационного имени не обязательно, и наоборот. Если указаны и то, и другое, то используется значение идентификатора пользователя. Если ни одно из полей не заполнено, сообщение не может быть отослано.

3. В поле “Subject” вводится тема сообщения. Размер строки темы сообщения ограничен 80 символами. Более длинные темы будут обрезаны на 80 символах. Кроме того, в таблице сообщений отображаются только первые 16 символов темы.
4. В данном поле вводится текст сообщения. Максимальный размер текста сообщения, посылаемого привилегированным пользователем с помощью CGI-программ **master** или **judge** ограничен максимальным размером пакета в протоколе взаимодействия программ **master** или **judge** и **serve** (в настоящее время — 128 Кб) и максимальным размером данных, принимаемых CGI-программами от браузера (в настоящее время — ? Кб).
5. Сообщение отсылается при нажатии на кнопку “Send”.
6. Ссылки быстрой навигации подробно описаны [выше](#).

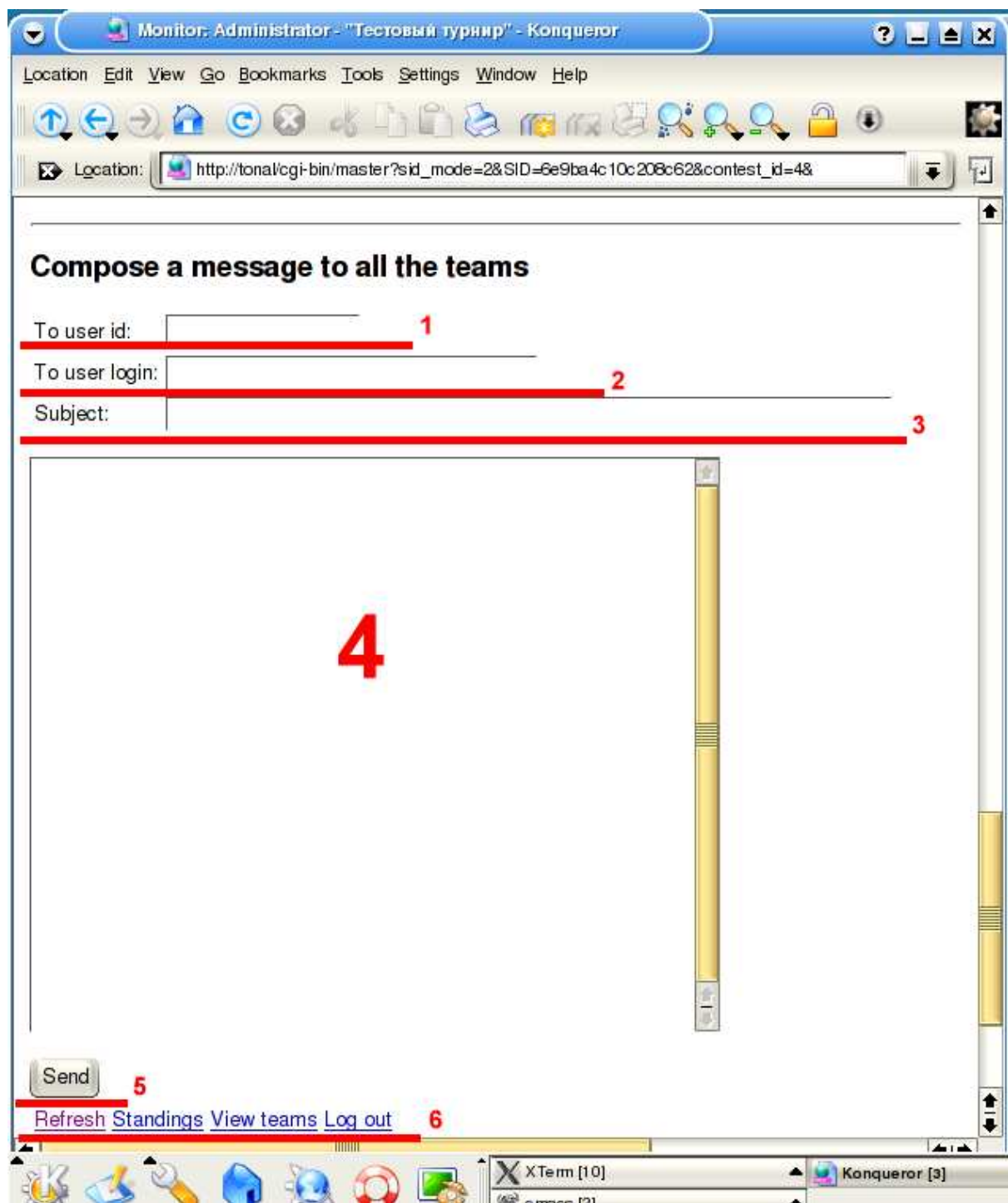


Рис. 4.10: Элементы интерфейса отправки сообщений

### 4.1.11 Выражение фильтра посылок

Выражение фильтра посылок позволяет задать условия, которым должны удовлетворять параметры посылок, чтобы быть показанными в списке посылок, генерируемом CGI-программами **master** и **judge**. Выражение фильтра должно иметь тип `bool`, а в противном случае генерируется ошибка.

При генерации страницы привилегированного пользователя в CGI-программе **judge** или **master** выражение фильтра используется следующим образом. Выражение фильтра применяется ко всем записям в базе посылок, и если на некоторой записи результат вычисления выражения равен `true`, то данная запись помечается как удовлетворяющая фильтру. Если при вычислении выражения фильтра хотя бы для одной посылки возникла ошибка, отображается информация об ошибке, а не результаты фильтрации. Если при вычислении выражения фильтра для всех записей завершилось успешно, из записей, удовлетворяющих выражению фильтра, отображаются записи в соответствии с заданным номером первой и номером последней записей.

Выражение фильтра посылок имеет Си-подобный синтаксис. Пробельные символы, разделяющие элементы выражения, игнорируются. В именах типов и полей регистр символов является значимым.

#### Типы данных

Выражение фильтра посылок строго типизировано. Выражение может состоять из операций над значениями следующих типов:

- **Булевский.** Булевский тип имеет имя `bool`. Литералы булевского типа обозначаются `false` и `true` соответственно.
- **Целый.** Целый тип имеет имя `int`. В текущей реализации целый тип является 32-битным знаковым типом. Литералы целого типа могут записываться в восьмеричной, десятичной и шестнадцатеричной системах счисления как в языке Си. Знак «минус» перед числом не является частью числа, а рассматривается как унарная операция изменения знака.
- **Строковый.** Строковый тип имеет имя `string`. Литералы строкового типа записываются либо в кавычках (`"`), либо в апострофах (`'`). Литералы первого вида не могут содержать внутри себя символ кавычки, а литералы второго вида — символ апострофа.
- **Календарный.** Календарный тип имеет имя `date_t` и предназначен для записи астрономического времени. Литералов календарного типа не существует. Астрономическое время при расчётах представляется в формате **POSIX**, то есть как количество секунд, прошедших с 1 января 1970 года.
- **Длительностный.** Длительностный тип имеет имя `dur_t` и предназначен для записи длительностей. При расчётах длительностный тип рассматривается как целый тип, и все длительности представляются в секундах. Литералов длительностного типа не существует.
- **Размерный.** Размерный тип имеет имя `size_t` и предназначен для представления размеров объектов. Литералов размерного типа не существует. При расчётах размерный тип рассматривается как беззнаковый целый тип.



	bool	int	string	date_t	dur_t	size_t	result_t	hash_t	ip_t
bool	+	+	+						
int	+	+	+	+	+	+	+		+
string	+	+	+	+	+	+	+	+	+
date_t	+	+	+	+					
dur_t	+	+	+		+				
size_t	+	+	+			+			
result_t	+	+	+				+		
hash_t	+	+	+					+	
ip_t	+	+	+						+

Таблица 4.1: Допустимые комбинации типов в операции преобразования типов

- **Статусный.** Статусный тип имеет имя `result_t` и предназначен для представления статуса посылки. Статусный тип — это перечислимый тип, состоящий из литералов OK, CE, RT, TL, PE, WA, CF, PT, AC, IG, RU, CD, CG, AV, EM (см. таблицу 5.1).
- **Контрольный.** Контрольный тип имеет имя `hash_t` и предназначен для представления значений контрольной суммы SHA1. Литералов контрольного типа не существует.
- **Адресный.** Адресный тип имеет имя `ip_t` и предназначен для представления IP-адресов. Литералов адресного типа не существует.

## Преобразования типов

Выражение фильтра посылок строго типизировано, но для преобразований значений одного типа в значения другого типа могут использоваться операции преобразования типа. Операция преобразования типа записывается в стиле Си++, то есть следующим образом:

```
type-cast-op = type-name "(" expr ") "
```

Здесь `type-name` — имя типа, одно из перечисленных выше `bool`, `int`, `string`, `date_t`, `dur_t`, `size_t`, `result_t`, `hash_t`, `ip_t`. Не все возможные преобразования типов допустимы. Возможные комбинации исходного типа и целевого типа показаны в таблице 4.1. В строках таблицы записан исходный тип, а в столбцах таблицы — целевой тип.

Рассмотрим подробнее каждое нетривиальное преобразование типов.

- **int к bool.** Даёт результат `false`, если приводимое целое значение равно нулю, и `true` иначе.
- **string к bool.** Даёт результат `false`, если значение строки равно `false` (без учёта регистра), и результат `true`, если значение строки равно `true` (без учёта регистра). Если значение строки не может быть распознано как булевский литерал, генерируется ошибка вычисления выражения.
- **date\_t к bool.** Даёт результат `false`, если значение астрономического времени во внутреннем формате равно нулю, и `true` иначе. На практике астрономическое время может быть равно нулю (то есть 0:00:00 1 января 1970 года по Гринвичу), когда значение соответствующей переменной или поля не установлено.

- **dur\_t** к **bool**. Даёт результат `false`, если значение длительности равно 0 секундам, и `true` иначе.
- **size\_t** к **bool**. Даёт результат `false`, если значение размера равно 0 байтам, и `true` иначе.
- **result\_t** к **bool**. Даёт результат `true`, если значение статуса равно OK, и `false` иначе. Обратите внимание на отличие семантики этой операции от других операций преобразования в булевский тип.
- **hash\_t** к **bool**. Даёт результат `false`, если значение хэш-кода равно нулю, то есть все 20 байт хэш-кода равны нулю, и `true` иначе. Нулевой хэш-код означает, что значение соответствующего поля или переменной не установлено.
- **ip\_t** к **bool**. Даёт результат `false`, если значение IP-адреса равно нулю, то есть все 4 байта IP-адреса нулевые, и `true` иначе. Нулевой IP-адрес означает, что значение соответствующего поля или переменной не установлено.
- **bool** к **int**. Булевское значение `false` преобразуется в целое значение 0, а булевское значение `true` — в целое значение 1.
- **string** к **int**. Строка рассматривается как запись числа в восьмеричной, десятичной или шестнадцатеричной нотации (в синтаксисе Си), возможно со знаком. Если число записано корректно и при переводе в 32-битный знаковый тип не возникает переполнения, результатом операции является соответствующее число. В противном случае генерируется ошибка.
- **date\_t** к **int**. Поскольку в настоящее время внутреннее представление типа `date_t` — это 32-битное знаковое число, то значением такого преобразования типов будет внутреннее представление астрономического времени.
- **dur\_t** к **int**. Поскольку в настоящее время внутреннее представление типа `dur_t` — это 32-битное знаковое число, то значением такого преобразования типов будет внутреннее представление промежутка времени, то есть число секунд в нём.
- **size\_t** к **int**. В настоящее время тип `size_t` — это 32-битный беззнаковый целый тип. Если значение размера при переводе из беззнакового представления в знаковое не вызывает переполнение (то есть значение размера не превосходит  $2^{31} - 1$ , выполняется преобразование из беззнакового типа в знаковый без изменения представления числа. В противном случае генерируется ошибка.
- **result\_t** к **int**. Результатом такого преобразования является числовое значение, соответствующее коду статуса (см. таблицу 5.1).
- **hash\_t** к **int**. Результатом такого преобразования является целое значение, составленное из первых 4 байтов хэш-кода. Предполагается, что байты в числе записываются в порядке от младшего к старшему (little-endian).
- **ip\_t** к **int**. Поскольку IP-адрес (IPv4) состоит из 4 байт, эти 4 байта рассматриваются как целое число со знаком, записанное в порядке байт от старшего к младшему.

- **bool** к **string**. Значение `false` преобразовывается в строку `false`, а значение `true` — в строку `true`.
- **int** к **string**. Целое значение преобразуется в строку, содержащую запись десятичного целого числа со знаком.
- **date\_t** к **string**. Астрономическое время преобразуется в строку вида

`YYYY/MM/DD hh:mm:ss`

Здесь `YYYY` — год, `MM` — месяц (от 01 до 12), `DD` — день месяца (от 01 до 31), `hh` — час (от 00 до 23), `mm` — минута (от 00 до 59), `ss` — секунда (от 00 до 59).

- **dur\_t** к **string**. Продолжительность преобразуется в строку вида

`#h:mm:ss`

Здесь `#` — знак продолжительности (для положительной продолжительности знак не выводится, а для отрицательной печатается знак `-`), `h` — число полных часов, `mm` — число минут в последнем неполном часе (от 00 до 59), `ss` — число секунд в последней неполной минуте (от 00 до 59).

- **size\_t** к **string**. Значение размера преобразуется в строку, содержащую запись десятичного целого числа без знака.
- **result\_t** к **string**. Значение статуса преобразовывается в строку в соответствии с таблицей 5.1 (столбец «Выражение фильтра»). Например, значение статуса `RUN_TIME_ERR` преобразуется в строке `RT`.
- **hash\_t** к **string**. Значение хэш-кода распечатывается в виде 40 шестнадцатеричных цифр (по два символа на байт от первого байта хэш-кода к последнему). В качестве шестнадцатеричных цифр используются десятичные цифры и строчные латинские буквы от `a` до `f`.
- **ip\_t** к **string**. IP-адрес переводится в строку, содержащую запись адреса в стандартной точечной нотации (то есть 4 десятичных числа от 0 до 255, разделённые точками).
- **int** к **date\_t**. Поскольку тип `date_t` представляется как 32-битное целое знаковое число, при данном преобразовании целое значение рассматривается как внутреннее представление астрономического времени.
- **string** к **date\_t**. Для преобразования строки к типу астрономического времени строка должна иметь значение, удовлетворяющее следующему формату

`Y / M / D [ h [ : m [ : s ] ] ]`

Здесь запись `[ X ]`, означает, что элемент `X` может отсутствовать, символы `/` и `:` представляют сами себя, а `Y`, `M`, `D`, `h`, `m`, `s` — это целые десятичные числа. Они могут отделяться от `/` и `:` произвольным количеством пробельных символов. `Y` — это год, который должен находиться в интервале от 1900 до 2100, `M` — порядковый номер месяца (от 1 до 12), `D` — порядковый номер дня в месяце (от 1 до 31), `h` — час суток (от 0

до 23), *m* — минута (от 0 до 59), *s* — секунда (от 0 до 59). Если часы, минуты или секунды опущены, они полагаются равными 0. Кроме того, дата должна находиться в диапазоне представимых в POSIX дат (то есть примерно от 1906 до 2038 года). Если все условия выполняются, строковая запись астрономического времени конвертируется во внутреннее представление, а в противном случае генерируется ошибка.

- **int** к **dur\_t**. Поскольку тип **dur\_t** представляется как 32-битное целое знаковое число, при данном преобразовании целое значение рассматривается как внутреннее представление промежутка времени.
- **string** к **dur\_t**. Для преобразования строки к длительностному значению строка должна иметь значение, удовлетворяющее следующему формату.

[ # ] h [ : m [ : s ] ]

Здесь запись [ *X* ], означает, что элемент *X* может отсутствовать. # — это знак + или -. *h*, *m* и *s* — положительные целые числа. *h* — количество часов, *m* — количество минут (от 0 до 59), *s* — количество секунд (от 0 до 59). Если минуты или секунды опущены, их число полагается равным 0. После перевода в секунды длительность должна находиться в диапазоне целых чисел, представимых 32-битными знаковыми числами. Если все условия выполняются, то строковая запись длительностного значения конвертируется во внутреннее представление, а в противном случае генерируется ошибка.

- **int** к **size\_t**. Поскольку **int** — это 32-битный знаковый тип, а **size\_t** — 32-битный беззнаковый тип, неотрицательные значения типа **int** преобразовываются в значения типа **size\_t** тривиально. При попытке преобразования отрицательных чисел генерируется ошибка.
- **string** к **size\_t**. Строка должна содержать запись беззнакового целого числа в восьмеричной, десятичной или шестнадцатеричной системе (как в языке Си). Число должно быть представимо 32-битным беззнаковым числом. Если эти условия выполняются, происходит преобразование строки в значение типа **size\_t**, а в противном случае генерируется ошибка.
- **int** к **result\_t**. Целое значение должно быть равным одному из значений статусов, перечисленных в таблице 5.1. В этом случае выполняется преобразование типа, а иначе генерируется ошибка.
- **string** к **result\_t**. Строка должна быть равна одной из строк, перечисленных в таблице 5.1 в столбце «Выражение фильтра». В этом случае выполняется преобразование, а иначе генерируется ошибка.
- **string** к **hash\_t**. Строка должна содержать 40 шестнадцатеричных цифр (буквы от *a* до *f* могут быть как заглавными, так и строчными). Пробельные символы в начале и конце строки игнорируются. 40 шестнадцатеричных цифр конвертируются в 20-байтное значение хэш-кода. Предполагается, что порядок байт в слове — от младшего к старшему (little-endian).
- **int** к **ip\_t**. Поскольку значения типа **ip\_t** представляются 32-битными числами, преобразование из типа **int** не изменяет представления данных, то есть целое число рассматривается как запись IP-адреса во внутреннем представлении.

- **string** к **ip\_t**. Для преобразования строки к типу **ip\_t** строка должна иметь значение, удовлетворяющее следующему шаблону.

A.B.C.D

Здесь . — это символ «точка», а A, B, C и D — целые беззнаковые числа от 0 до 255. Если эти условия выполняются строка конвертируется в соответствующее значение типа **ip\_t**, а в противном случае генерируется ошибка.

## Примитивы доступа к базе посылок

Текущей посылкой назовём обрабатываемую при вычислении выражения фильтра посылку. На основании результата вычисления выражения фильтра определяется, может ли быть отображена текущая посылка.

Примитивы доступа к базе посылок позволяют получать значения полей как текущей посылки, так и любой другой посылки с заданным номером. Примитивы имеют следующий общий вид:

`f [ ( expr ) ]`

Здесь в квадратных скобках [ и ] записана необязательная часть. Круглые скобки ( и ) обозначают сами себя. `f` — это имя примитива, а `expr` — это выражение целого типа. Если выражение в скобках не задано, то берётся значение поля, соответствующего имени примитива `f`, текущей посылки. Если выражение в скобках не является выражением целого типа, генерируется ошибка. В противном случае выражение вычисляется, и пусть при вычислении выражения ошибок не возникло, а  $n$  — результат вычисления выражения.

Если  $n < 0$ , то  $n$  задаёт номер посылки начиная от последней, то есть  $n' = N + n$ , где  $N$  — общее количество посылок в базе посылок в данный момент. В противном случае предполагаем, что  $n$  задаёт номер посылки начиная от нулевой, то есть  $n' = n$ . Если теперь  $n' < 0$  или  $n' \geq N$ , то генерируется ошибка вычисления выражения фильтра. Если же  $n'$  удовлетворяет ограничениям, берётся значение поля, соответствующего имени примитива `f`, в посылке с номером  $n'$ .

Примитивы доступа к базе посылок перечислены в таблице [4.2](#).

## Другие примитивы

В выражении фильтра можно использовать примитивы для получения текущего времени, текущей продолжительности турнира и некоторые другие. Все такие примитивы рассмотрены в данном разделе. Примитивы имеют следующий общий вид:

`p`

где `p` — это имя примитива. Примитивы перечислены в таблице [4.3](#).

## Приоритеты операций

В данном разделе перечислены все операции, допустимые в выражении фильтра. Операции перечисляются в порядке убывания приоритета от наиболее приоритетных к наименее приоритетным.

Имя примитива	Тип значения	Описание
time	time_t	Астрономическое время получения посылки сервером (значение поля timestamp)
dur	dur_t	Длительность времени от начала турнира до получения посылки сервером
size	size_t	Размер исходного текста программы в байтах (значение поля size).
hash	hash_t	Хэш-код текста программы (значение поля sha1).
ip	ip_t	IP-адрес клиента, на котором работает веб-браузер, и с которого была получена данная посылка (значение поля ip).
uid	int	Идентификатор пользователя, от имени которого была выполнена данная посылка (значение поля team). Идентификатор пользователя всегда больше нуля.
login	string	Регистрационное имя (login) пользователя, от имени которого была выполнена данная посылка.
lang	string	Короткое имя языка программирования, который был использован в данной посылке. Идентификатор языка программирования находится в поле language записи базы посылок, а короткое имя языка программирования берётся из значения конфигурационной переменной <code>short_name</code> секции описания языка программирования файла конфигурации турнира <code>serve.cfg</code> .
prob	string	Короткое имя задачи, решаемой в данной посылке. Идентификатор задачи находится в поле problem записи базы посылок, а короткое имя задачи берётся из значения конфигурационной переменной <code>short_name</code> секции описания задачи файла конфигурации турнира <code>serve.cfg</code> .
result	result_t	Статус посылки (значение поля status).
status	result_t	То же самое.
score	int	Балл за данную посылку без учёта штрафных баллов за посылки (значение поля score).
test	int	Минимальный номер теста, на котором программа дала неверный результат, или количество успешно пройденных тестов в зависимости от типа турнира.
imported	bool	Флаг импортированной посылки (значение поля is_imported).
hidden	bool	Флаг скрытой посылки (значение поля is_hidden).
readonly	bool	Флаг неизменяемой посылки (значение поля is_readonly).
variant	int	Действительный вариант. Если для вариантной задачи номер варианта, хранящийся в базе посылок (значение поля variant), равен 0, используется номер варианта, установленный в файле карты вариантов, устанавливаемом глобальной конфигурационной переменной <code>variant_map_file</code> конфигурационного файла описания турнира <code>serve.cfg</code> .
rawvariant	int	Хранимый вариант задачи в посылке (значение поля variant).

Таблица 4.2: Примитивы доступа к базе посылок

Имя примитива	Тип значения	Описание
id	int	Номер текущей посылки.
now	time_t	Текущее астрономическое время.
start	time_t	Астрономическое время начала турнира.
finish	time_t	Астрономическое время завершения турнира.
total	int	Общее количество записей в базе посылок.

Таблица 4.3: Прочие примитивы

- Все литеральные значения (булевские, целые, строковые, статусные), все примитивы, операции преобразования типа, операция записи выражения в скобках (*expr*) .
- Префиксные операции  $\sim$ ,  $!$ ,  $-$ ,  $+$ . Префиксные операции читаются справа налево.
- Мультипликативные операции  $*$ ,  $/$ ,  $\%$ . Мультипликативные операции левоассоциативны, то есть читаются слева направо.
- Аддитивные операции  $+$ ,  $-$ . Они читаются также слева направо.
- Операции сдвига  $<<$  и  $>>$ . Читаются слева направо.
- Операции отношений  $==$ ,  $!=$ ,  $<$ ,  $>$ ,  $<=$ ,  $>=$ . Читаются слева направо.
- Операция побитового «и»  $\&$ . Читается слева направо.
- Операция побитового исключающего «или»  $\wedge$ . Читается слева направо.
- Операция побитового «или»  $|$ . Читается слева направо.
- Операция логического «и»  $\&\&$  или `and` (эти формы записи синонимичны). Читается слева направо.
- Операция логического «или»  $||$  или `or` (эти формы записи синонимичны). Читается слева направо.

Далее рассмотрим все перечисленные операции.

### Операция побитового отрицания $\sim$

Операция применима только к значению типа `int`. Результат операции — значение типа `int`, получаемое побитовым отрицанием первого значения.

### Операция логического отрицания $!$

Операция применима только к значению типа `bool`. Результат операции — значение типа `bool`, получаемое логическим отрицанием исходного значения.

## Операция смены знака –

Унарная операция смены знака применима только к значению типа `int`. Результат операции — значение типа `int` с противоположным знаком. Если результат операции не представим в типе `int` (то есть 32-битным знаковым числом), диагностируется ошибка переполнения.

## «Пустая» операция +

Данная операция применима только к значению типа `int` и даёт результат типа `int`. Результирующее значение равно исходному.

## Операция умножения \*

Для операции умножения возможны следующие комбинации типов операндов.

- Оба аргумента имеют тип `int`. Результат имеет тип `int`. Если результат не представим в типе `int` диагностируется переполнение.
- Один аргумент имеет тип `int`, а другой — `dur_t`. В этом случае результат имеет тип `dur_t`. Если произведение аргументов не представимо в типе `dur_t`, диагностируется переполнение.
- Один аргумент имеет тип `int`, а другой — `size_t`. Значение аргумента целого типа в данном случае должно быть больше нуля. Произведение имеет тип `size_t`. Если значение произведения не представимо в типе `size_t`, диагностируется переполнение.

## Операция целочисленного деления /

Для операции целочисленного деления возможны следующие комбинации типов операндов. Если делитель равен нулю, диагностируется ошибка деления на ноль. Если результат операции не представим в типе результата, диагностируется ошибка переполнения. Например, при делении минимального значения типа `int`, равного  $-2^{31}$  на  $-1$  получается значение  $2^{31}$ , не представимое в типе `int`.

- Оба аргумента имеют тип `int`. Тогда результат имеет тип `int`.
- Оба аргумента имеют тип `dur_t`. Тогда результат имеет тип `int`.
- Оба аргумента имеют тип `size_t`. Тогда результат имеет тип `int`.
- Делимое имеет тип `dur_t`, а делитель — тип `int`. Тогда результат имеет тип `dur_t`.
- Делимое имеет тип `size_t`, а делитель — тип `int`. Результат в этом случае имеет тип `size_t`. Делитель должен быть положителен.



## Операция взятия остатка от деления %

Операция взятия остатка поддерживает несколько комбинаций типов аргументов. Если делитель равен нулю, диагностируется ошибка деления на ноль. Если тип делителя — какой-либо знаковый тип (например, `int` или `dur_t`), и делитель отрицателен, диагностируется ошибка неверного аргумента. Если результат операции не представим в типе результата, диагностируется ошибка переполнения. Допустимые комбинации типов операндов перечислены ниже.

- Оба аргумента имеют тип `int`. Тогда результат имеет тип `int`.
- Оба аргумента имеют тип `dur_t`. Тогда результат имеет тип `int`.
- Оба аргумента имеют тип `size_t`. Тогда результат имеет тип `int`.
- Делимое имеет тип `dur_t`, а делитель — тип `int`. Тогда результат имеет тип `dur_t`.
- Делимое имеет тип `size_t`, а делитель — тип `int`. Результат в этом случае имеет тип `size_t`.

## Операция сложения +

Операция сложения применима ко многим комбинациям типов операндов. В любом случае, если результат операции не представим в типе результата, генерируется ошибка переполнения. Допустимые комбинации типов операндов перечислены ниже.

- Оба аргумента имеют тип `int`. Результат также имеет тип `int`.
- Оба аргумента имеют тип `string`. Результат имеет тип `string` и равен конкатенации двух строк.
- Оба аргумента имеют тип `dur_t`. Результат также имеет тип `dur_t`.
- Оба аргумента имеют тип `size_t`. Результат также имеет тип `size_t`.
- Один аргумент имеет тип `int`, а другой — `time_t`. Результат имеет тип `time_t`.
- Один аргумент имеет тип `int`, а другой — `dur_t`. Результат имеет тип `dur_t`.
- Один аргумент имеет тип `time_t`, а другой — `dur_t`. Результат имеет тип `time_t`.

## Операция вычитания -

Операция вычитания применима ко многим комбинациям типов операндов. В любом случае, если результат операции не представим в типе результата, генерируется ошибка переполнения. Допустимые комбинации типов операндов перечислены ниже.

- Оба аргумента операции имеют тип `int`. Результат также имеет тип `int`.
- Первый аргумент операции имеет тип `int`, а второй — `dur_t`. Результат операции имеет тип `dur_t`.

- Первый аргумент операции имеет тип `time_t`, а второй — `int`. Результат операции имеет тип `time_t`.
- Оба аргумента операции имеют тип `time_t`. В этом случае результат имеет тип `dur_t`.
- Первый аргумент операции имеет тип `time_t`, а второй — `dur_t`. Результат имеет тип `time_t`.
- Первый аргумент имеет тип `dur_t`, а второй — `int`. Результат имеет тип `dur_t`.
- Оба аргумента операции имеют тип `dur_t`. В этом случае результат имеет тип `dur_t`.
- Первый аргумент операции имеет тип `size_t`, а второй — `int`. Тогда результат имеет тип `size_t`.
- Оба аргумента операции имеют тип `size_t`. В этом случае результат имеет тип `int`.

### Операции сдвига << и >>

Операции побитового сдвига влево << и побитового сдвига вправо >> применимы только к операндам типа `int`. Результат операции имеет тип `int`. Второй аргумент операции (количество бит сдвига) должен находиться в отрезке от 0 до 32 включительно. Если это условие не выполняется, генерируется ошибка недопустимого аргумента операции. Сдвиг вправо выполняется как побитовый сдвиг, то есть на место старших разрядов числа помещаются нулевые биты. Переполнения при сдвиге влево не диагностируется.

### Операции отношения == и !=

Операции проверки на равенство или неравенство применимы ко всем типам данных. Оба аргумента должны иметь один и тот же тип. Неявных преобразований типа не производится. Результат операции имеет тип `bool`.

### Операции отношения <, >, <=, >=

Данные операции применимы ко всем типам данных, кроме `result_t`, `hash_t` и `ip_t`. Оба аргумента операции должны иметь один и тот же тип данных. Результат операции имеет тип `bool`.

### Побитовые операции ^, &, |

Побитовые операции применимы только к аргументам типа `int` и в результате дают значение типа `int`.

### Логические операции && и ||

Логические операции применимы только к аргументам типа `bool` и дают в результате значение типа `bool`. Как и в языке Си значение второго аргумента операции не вычисляется, если значение всего выражение известно.

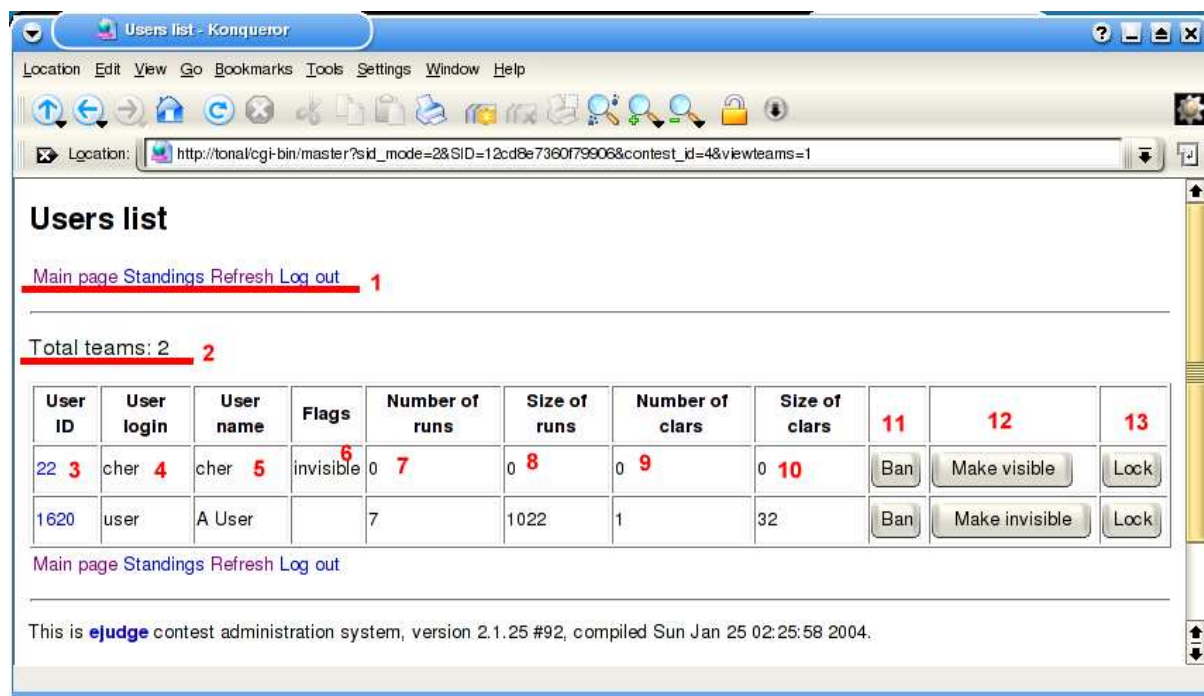


Рис. 4.11: Интерфейс управления участниками турнира

#### 4.1.12 Интерфейс управления участниками турнира

На рис. 4.11 приведён вид элементов управления участниками турнира, доступный с помощью CGI-программы администратора турнира `master`. Для просмотра и управления участниками турнира в главной странице управления турниром, а также на страницах просмотра текста программы решения, протокола тестирования, текущих результатов тестирования и сообщений участников нажать на ссылку “View teams”.

Чтобы пользователь имел полномочия просматривать список участников турнира, в конфигурационном файле турнира `contest.xml` в элементе `cap` для него должен быть установлен бит `LIST_CONTEST_USERS`.

Элементы управления участниками турнира рассмотрены ниже.

1. Ссылки быстрой навигации, доступные на данной странице, немного отличаются от ссылок быстрой навигации на главной странице.
  - Ссылка “Main page” вызывает переход на главную страницу управления турниром.
  - Ссылка “Standings” вызывает переход к таблице текущих результатов турнира.
  - Ссылка “Refresh” вызывает обновление данной страницы.
  - Нажатие на ссылку “Log out” приводит к завершению сеанса работы пользователя с программой `master`.
2. В данной строке отображается общее количество команд, зарегистрировавшихся на турнир.
3. Элементы управления 3–13 — это столбцы в таблице пользователей турнира.

В столбце “User ID” отображаются идентификаторы участников турнира. Идентификатор участника — это целое число, большее 0. Для каждого участника идентификатор в таблице представлен как гиперссылка. При нажатии на эту гиперссылку происходит возврат на главную страницу CGI-программы **master**, и при этом устанавливается выражение фильтра вида “uid == I”, где I — это идентификатор текущего участника. Такое выражение фильтра отбирает только послышки данного участника.

4. В столбце “User Login” отображается регистрационное имя пользователя.
5. В столбце “User Name” отображается имя пользователя. Имя пользователя устанавливается самим пользователем при регистрации на турнир.
6. В столбце “Flags” отображаются флаги, установленные для данного участника. Для участника могут быть установлены три флага в любой комбинации.
  - Флаг **banned**. Если для пользователя установлен этот флаг, участие пользователя в данном турнире запрещено администратором. Пользователь не сможет войти на страницу сдачи решений, его результаты не отображаются в таблице текущих результатов турнира.
  - Флаг **invisible**. Если для пользователя установлен данный флаг, результаты пользователя не отображаются в таблице текущих результатов. Участник, у которого установлен этот флаг, может использовать CGI-программы **master**, **judge** или **team**, может посылать решения и т. д. Рекомендуется устанавливать флаг “invisible” для привилегированных пользователей турнира (администратора, судей и т. д.).
  - Флаг **locked**. Пользователь, у которого установлен данный флаг, не может использовать программы **master**, **judge** или **team**, то есть не может сдавать решения. Однако, результаты пользователя отображаются в таблице текущих результатов турнира. Данный флаг удобно устанавливать у пользователей, составляющих неvirtуальную часть таблицы результатов виртуального турнира.
7. В столбце “Number of runs” отображается общее количество посылок, выполненных данным пользователем в данном турнире на текущий момент. Если это число равно 0, значит пользователь ещё не посылал решения на проверку. Обратите внимание, что для непривилегированных посылок действует ограничение на максимальное количество посылок, которое устанавливается глобальной конфигурационной переменной `max_run_num` конфигурационного файла описания турнира `serve.cfg`.
8. В столбце “Size of runs” отображается суммарный размер всех программ, переданных данным пользователем на проверку в данном турнире на текущий момент. Обратите внимание, что для непривилегированных посылок действует ограничение на максимальный суммарный размер посылок, которое устанавливается глобальной конфигурационной переменной `max_run_total` конфигурационного файла описания турнира `serve.cfg`.
9. В столбце “Number of clars” отображается общее количество сообщений, посланных данным пользователем в данном турнире на текущий момент. Для непривилегированных пользователей действует ограничение на максимальное количество сообщений жюри, которое устанавливается глобальной конфигурационной переменной `max_clar_num` конфигурационного файла описания турнира `serve.cfg`.

10. В столбце “Size of clars” отображается суммарный размер всех сообщений, посланных данным пользователем на текущий момент времени. Для непривилегированных пользователей действует ограничение на максимальный суммарный размер сообщений жюри, которое устанавливается глобальной конфигурационной переменной `max_clar_total` конфигурационного файла описания турнира `serve.cfg`.
11. С помощью кнопки “Ban”/“Unban” можно установить или сбросить флаг “banned” для данного пользователя. Если у пользователя флаг “banned” не установлен, данная кнопка отображается в виде “Ban”, и при нажатии на эту кнопку устанавливается флаг “banned”. Если у пользователя флаг “banned” уже установлен, данная кнопка отображается в виде “Unban”, и при нажатии на эту кнопку флаг “banned” сбрасывается.
12. С помощью кнопки “Make invisible”/“Make visible” можно установить или сбросить флаг “invisible” для данного пользователя. Если у пользователя флаг “invisible” не установлен, данная кнопка отображается в виде “Make invisible”, и при нажатии на эту кнопку устанавливается флаг “invisible”. Если у пользователя флаг “invisible” уже установлен, данная кнопка отображается в виде “Make visible”, и при нажатии на эту кнопку флаг “invisible” сбрасывается.
13. С помощью кнопки “Lock”/“Unlock” можно установить или сбросить флаг “locked” для данного пользователя. Если у пользователя флаг “locked” не установлен, данная кнопка отображается в виде “Lock”, и при нажатии на эту кнопку устанавливается флаг “locked”. Если у пользователя флаг “locked” уже установлен, данная кнопка отображается в виде “Unlock”, и при нажатии на эту кнопку флаг “locked” сбрасывается.

#### 4.1.13 Интерфейс редактирования посылок

На рис. 4.12 показан вид (верхняя часть) экрана редактирования информации о посылке. К редактированию посылки можно перейти, нажав на ссылку “View” некоторой посылки на главном экране программы **master**. Далее рассматриваются элементы интерфейса страницы редактирования посылок.

1. Ссылки быстрой навигации позволяют переключиться на другой экран программы **master**. Доступны следующие ссылки быстрой навигации:
  - “Main page” вызывает переход на главную страницу программы **master**.
  - “Standings” вызывает переход на страницу просмотра текущих результатов турнира.
  - “View teams” вызывает переход на страницу просмотра информации об участниках турнира.
  - “Refresh” вызывает обновление данной страницы.
  - “View report” вызывает переход к странице просмотра протокола тестирования для текущей посылки.
  - “View team report” вызывает переход к странице просмотра протокола тестирования для текущей посылки, предназначенного для участника турнира. Данная ссылка отображается, только если глобальная конфигурационная переменная `team_enable_rep_view` конфигурационного файла `serve.cfg` установлена в *true*.

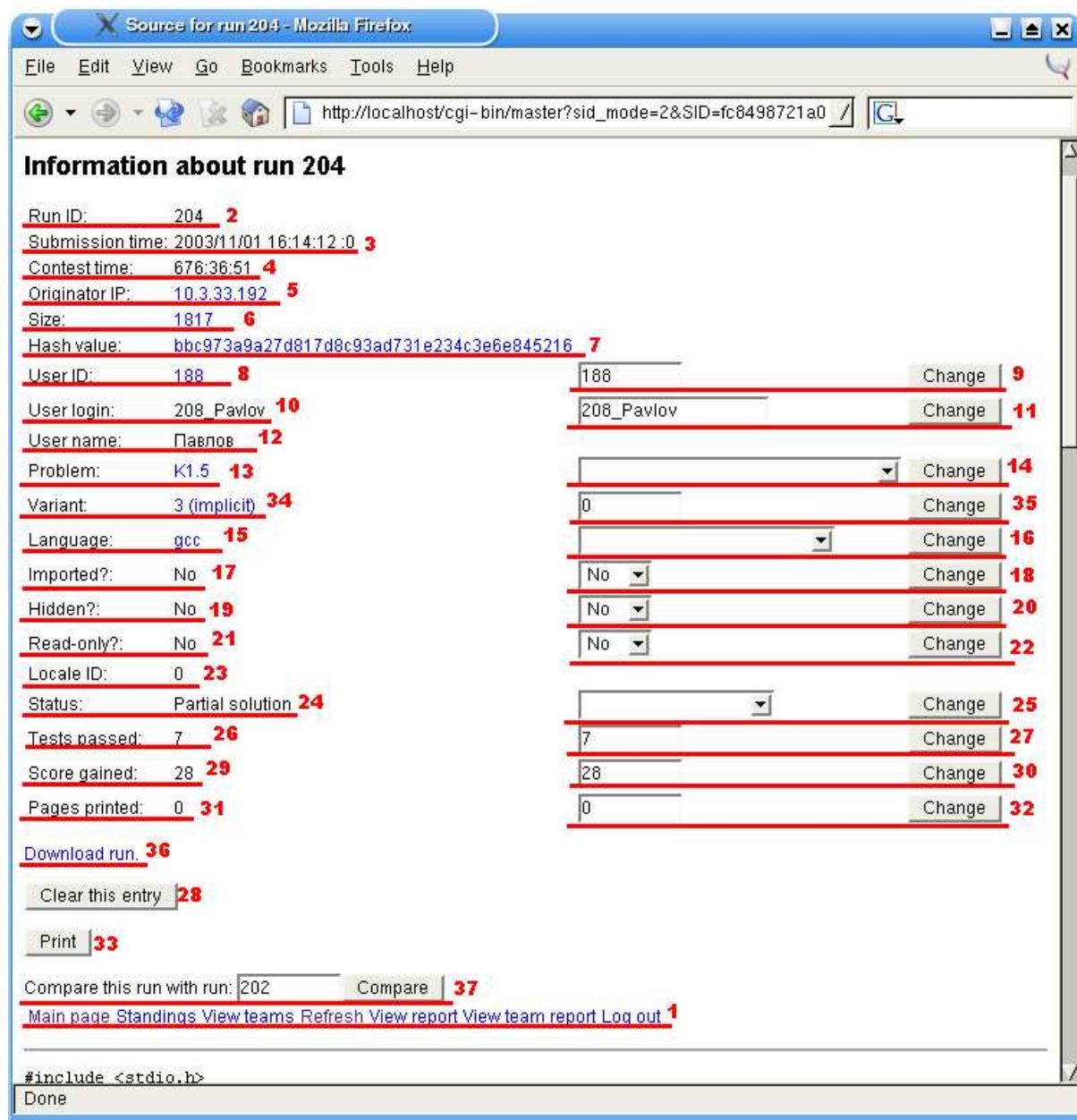


Рис. 4.12: Интерфейс редактирования посылок

- “Log out” приводит к выходу пользователя из программы **master** и завершению сессии работы.
2. В поле “Run ID” отображается идентификатор посылки. Первая (самая ранняя) посылка имеет идентификатор 0. Текущая версия системы **ejudge** имеет ограничение на максимальное количество посылок, равное 100000. См. также описание поля [submission](#) журнала посылок [run.log](#).
  3. Поле “Submission time” показывает астрономическое время, когда была выполнена данная посылка, точнее, когда она была получена сервером турнира **serve**. Время отображается в формате YYYY/MM/DD hh:mm:ss, где YYYY — год, MM — номер месяца в году (от 1 до 12), DD — день месяца (от 1 до 31), hh — часы (от 0 до 23), mm — минуты, и ss — секунды. См. также описание поля [timestamp](#) журнала посылок [run.log](#).
  4. Поле “Contest time” показывает время от начала турнира до момента получения сервером турнира данной посылки. Время отображается в формате h:mm:ss, где h — количество часов, mm — минут, и ss — секунд.
  5. Поле “Originator IP” показывает IP-адрес клиента, с которого была выполнена посылка. IP-адрес показывается в стандартной точечной нотации. IP-адрес генерируется как ссылка, нажав на которую происходит возврат на основной экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки, выполненные с данного IP-адреса. См. также описание поля [ip](#) журнала посылок [run.log](#).
  6. Поле “Size” показывает размер исходного текста посылки в байтах. Размер показывается как ссылка, при нажатии на которую происходит возврат на главный экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки, у которых размер исходного текста совпадает с размером исходного текста данной посылки. См. также описание поля [size](#) журнала посылок [run.log](#).
  7. Поле “Hash value” показывает контрольную сумму исходного текста программы данной посылки. Контрольная сумма вычисляется по алгоритму SHA1 и представляется как 20-байтное число в шестнадцатеричном виде. Контрольная сумма показывается как гиперссылка, при нажатии на которую происходит возврат на главный экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки, у которых контрольная сумма исходного текста совпадает с контрольной суммой исходного текста данной посылки. См. также описание поля [sha1](#) журнала посылок [run.log](#).
  8. В поле “User ID” отображается числовой идентификатор участника турнира, выполнившего данную посылку. Числовые идентификаторы участников находятся в диапазоне от 1 до 100000. Идентификатор участника генерируется как гиперссылка, при нажатии на которую происходит возврат на главный экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки данного пользователя. См. также описание поля [team](#) журнала посылок [run.log](#).
  9. С помощью данного элемента ввода можно изменить идентификатор пользователя, приписываемый данной посылке. Для этого нужно ввести новый числовой идентификатор и нажать на кнопку “Change”. Пользователь с данным идентификатором должен



существовать и быть зарегистрированным на данный турнир (то есть иметь статус регистрации “OK”). Пользователь может иметь статус “Locked”, “Banned” или “Invisible” в данном турнире.

10. В поле “User login” отображается регистрационное имя участника турнира. Регистрационное имя используется участником при входе в программы системы **ejudge**. Как и идентификатор пользователя, регистрационное имя уникально для всех зарегистрированных пользователей. См. также описание поля [team](#) журнала посылки [run.log](#).
11. С помощью данного элемента ввода можно задать регистрационное имя пользователя, которому будет приписана данная посылка. Изменение регистрационного имени пользователя приведёт к изменению идентификатора пользователя (и наоборот). Для этого нужно ввести новое регистрационное имя пользователя и нажать на кнопку “Change”. Пользователь с данным идентификатором должен существовать и быть зарегистрированным на данный турнир (то есть иметь статус регистрации “OK”). Пользователь может иметь статус “Locked”, “Banned” или “Invisible” в данном турнире.
12. В поле “User name” отображается имя пользователя. Имя пользователя выбирается пользователем при регистрации на турнир с помощью программы **register**, а также может быть установлено администратором турнира с помощью программы **edit-userlist**. Несколько пользователей могут установить себе одно и то же имя пользователя, однако каждый пользователь может иметь только одно имя. См. также описание поля [team](#) журнала посылки [run.log](#).
13. В поле “Problem” отображается короткое имя задачи (см. конфигурационную переменную [short\\_name](#) секции описания задачи конфигурационного файла турнира [serve.cfg](#)), по которой была выполнена данная посылка. Короткое имя задачи генерируется как гиперссылка, при нажатии на которую происходит возврат на главный экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки по данной задаче. См. также описание поля [problem](#) журнала посылки [run.log](#).
14. С помощью данного диалогового элемента можно изменить задачу данной посылки. Для этого в выпадающем меню нужно выбрать новую задачу и нажать на кнопку “Change”. В выпадающем меню выбора задачи отображается как короткое имя, так и полное имя задачи.
15. В поле “Language” отображается короткое имя языка программирования (см. конфигурационную переменную [short\\_name](#) секции описания языка программирования конфигурационного файла турнира [serve.cfg](#)), который был использован в данной посылке. Короткое имя языка программирования генерируется как гиперссылка, при нажатии на которую происходит возврат на главный экран программы **master** с одновременной установкой фильтра посылок таким образом, чтобы отображались только посылки с таким же языком программирования. См. также описание поля [language](#) журнала посылки [run.log](#).
16. С помощью данного диалогового элемента можно изменить язык программирования данной посылки. Для этого в выпадающем меню нужно выбрать новый язык программирования и нажать на кнопку “Change”. В выпадающем меню выбора языка программирования отображается как короткое имя, так и полное имя языка программирования.



17. В поле “Imported?” отображается текущее значение флага импортированной посылки. Флаг может принимать два значения: “No” (*false*) и “Yes” (*true*). См. также описание поля [is\\_imported](#) журнала посылок [run.log](#).
18. С помощью данного диалогового элемента можно изменить значение флага импортированной посылки. Для этого в выпадающем меню нужно выбрать новое значение и нажать на кнопку “Change”.
19. В поле “Hidden?” отображается текущее значение флага скрытой посылки. Флаг может принимать два значения: “No” (*false*) и “Yes” (*true*). См. также описание поля [is\\_hidden](#) журнала посылок [run.log](#).
20. С помощью данного диалогового элемента можно изменить значение флага скрытой посылки. Для этого в выпадающем меню нужно выбрать новое значение и нажать на кнопку “Change”.
21. В поле “Read-only?” отображается текущее значение флага неизменяемой посылки. Флаг может принимать два значения: “No” (*false*) и “Yes” (*true*). См. также описание поля [is\\_readonly](#) журнала посылок [run.log](#).
22. С помощью данного диалогового элемента можно изменить значение флага неизменяемой посылки. Для этого в выпадающем меню нужно выбрать новое значение и нажать на кнопку “Change”.
23. В поле “Locale ID” отображается идентификатор языкового окружения, установленный участником турнира в момент отправки им данной посылки. В настоящее время система **ejudge** поддерживает два языковых окружения: английское (идентификатор 0) и русское (идентификатор 1). См. также описание поля [locale\\_id](#) журнала посылок [run.log](#).
24. В поле “Status” отображается статус данной посылки. Все возможные статусы посылок приведены в таблице [5.1](#). См. также описание поля [status](#) журнала посылок [run.log](#).
25. С помощью данного диалогового элемента можно модифицировать статус текущей посылки. Для этого в выпадающем меню нужно выбрать новое значение статуса и нажать на кнопку “Change”.
26. В поле “Tests passed” отображается количество тестов, успешно пройденных решением (в режимах турнира *KIROV* и *OLYMPIAD*). Если статус текущей посылки таков, что запуск на тестирование программы не выполнялся, в данном поле отображается N/A.  
В режиме турнира *ACM* данное поле называется “Failed test” и показывает минимальный номер теста, на котором программа дала неверный результат. Если решение прошло все тесты или статус текущей посылки таков, что запуск на тестирование программы не выполнялся, в данном поле отображается N/A.  
См. также описание поля [tests](#) журнала посылок [run.log](#).
27. С помощью данного диалогового элемента можно изменить количество пройденных тестов или номер первого теста для данной посылки. Для этого нужно ввести новое значение и нажать на кнопку “Change”. Чтобы установить данное поле в значение «не определено» (N/A), в поле ввода нужно ввести число -1.

28. При нажатии на кнопку “Clear this entry” текущая запись будет очищена, включая все поля таблицы посылок и файлы с исходным текстом и протоколами тестирования. При нажатии на данную кнопку запрашивается подтверждение операции очистки.
29. В поле “Score gained” отображается количество баллов, полученное за данную посылку. Данное поле доступно только в режимах турнира *KIROV* или *OLYMPIAD*. Количество баллов отображается без учёта возможных штрафов за повторные попытки. Если статус текущей посылки таков, что запуск на тестирование программы не выполнялся, в данном поле отображается N/A. См. также описание поля `score` журнала посылок `run.log`.
30. С помощью данного диалогового элемента можно изменить количество баллов за решение. Данный диалоговый элемент доступен только в режимах турнира *KIROV* или *OLYMPIAD*. Для изменения значения поля нужно ввести новое значение и нажать на кнопку “Change”. Чтобы установить данное поле в значение «не определено» (N/A), в поле ввода нужно ввести число -1.
31. В поле “Pages” отображается количество страниц, использованных при распечатке пользователем текста программы данной посылки. Если пользователь не печатал текст, значение данного поля равно 0. Печать исходного текста посылки на принтере непривилегированным пользователем (из-под программы `team`) возможна только, если глобальная конфигурационная переменная `enable_printing` файла конфигурации турнира `serve.cfg` установлена в значение `true`. См. также описание поля `pages` журнала посылок `run.log`.
32. С помощью данного диалогового элемента можно изменить количество страниц распечатки для данной посылки. Для этого в поле ввода нужно задать новое значение и нажать на кнопку “Change”.
33. С помощью кнопки “Print” можно отправить исходный текст текущей посылки на принтер. Эта операция является привилегированной операцией, для которой требуется бит привилегий `PRINT_RUN`. В данном случае не действуют квоты печати, можно несколько раз напечатать одну и ту же программу. В результате привилегированной печати не изменяется поле `pages`, но есть непривилегированный пользователь, который выполнил данную посылку, всё равно сохраняет возможность её печати на принтер.
- Для печати посылок на принтер требуется задать корректные значения для конфигурационных переменных `a2ps_path`, `a2ps_args`, `lpr_path`, `lpr_args`.
34. В поле “Variant” отображается вариант задачи. Данное поле отображается только для вариантных задач, то есть для задач, у которых конфигурационная переменная `variant_num` файла конфигурации турнира `serve.cfg` установлена в значение, большее 0. Значение варианта изменяется от 1 и до значения, устанавливаемого этой конфигурационной переменной. Если в базе посылок номер варианта для данной посылки явно не установлен, то номер варианта определяется по таблице вариантов (см. конфигурационную переменную `variant_map_file` файла конфигурации турнира `serve.cfg`). В этом случае номер варианта отображается с пометкой “Implicit”.
- См. также описание поля `variant` журнала посылок `run.log`.
35. С помощью поля редактирования номера варианта можно изменить номер варианта данной посылки. Для этого в поле ввода нужно задать новый номер варианта и нажать

на кнопку “Change”. Чтобы сбросить явное задание номера варианта в базе посылок (то есть чтобы использовался файл задания вариантов для пользователей), в поле ввода нужно ввести номер варианта 0.

36. Ссылка “Download run” позволяет загрузить исходный текст посылки. Несмотря на то, что страница просмотра исходного текста уже содержит сам исходный текст, возможность загрузки исходного текста необходима по следующим причинам:

- Исходный текст можно сразу сохранить в файл, а не копировать его через буфер обмена, избегая таким образом искажения данных, возможных при копировании через буфер обмена.
- Для посылок по задачам, для которых конфигурационная переменная `binary` конфигурационного файла описания турнира `serve.cfg` установлена в `true`, исходный текст посылки не отображается, так как может содержать управляющие коды, и во многих случаях (например, в случае посылки архива исходных текстов) его отображение вообще бессмысленно. В этом случае ссылка “Download run” — единственный способ ознакомиться с исходным текстом посылки.

37. Данный элемент управления позволяет сравнить исходные тексты двух посылок. Текущая посылка сравнивается с посылкой, номер которой указывается в поле ввода. Для сравнения нужно ввести номер другой посылки и нажать на кнопку “Compare”. Сравнение ведётся с помощью программы `diff` командой `diff -u`, при этом указанная посылка будет первым аргументом программы (то есть «старым» файлом), а текущая посылка — вторым аргументом программы (то есть «новым» файлом). Путь к команде `diff` можно задать с помощью глобальной конфигурационной переменной `diff_path`. При отображении страницы просмотра исходного текста в поле ввода номера сравниваемой посылки записывается номер ближайшей (в сторону уменьшения) посылки, для которой значения идентификатора команды, идентификатора языка программирования и идентификатора задачи совпадают со значениями соответствующих полей текущей посылки. Другими словами, по умолчанию в поле ввода номера посылки вписывается предыдущая посылка того же участника по той же задаче.

#### 4.1.14 Просмотр исходного кода посылок

На рис. 4.13 представлены элементы управления, доступные на странице редактирования посылки.

- Ссылки быстрой навигации позволяют переключиться на другой экран программы **master**. Доступны следующие ссылки быстрой навигации:
  - “Main page” вызывает переход на главную страницу программы **master**.
  - “Standings” вызывает переход на страницу просмотра текущих результатов турнира.
  - “View teams” вызывает переход на страницу просмотра информации об участниках турнира.
  - “Refresh” вызывает обновление данной страницы.
  - “View report” вызывает переход к странице просмотра протокола тестирования для текущей посылки.

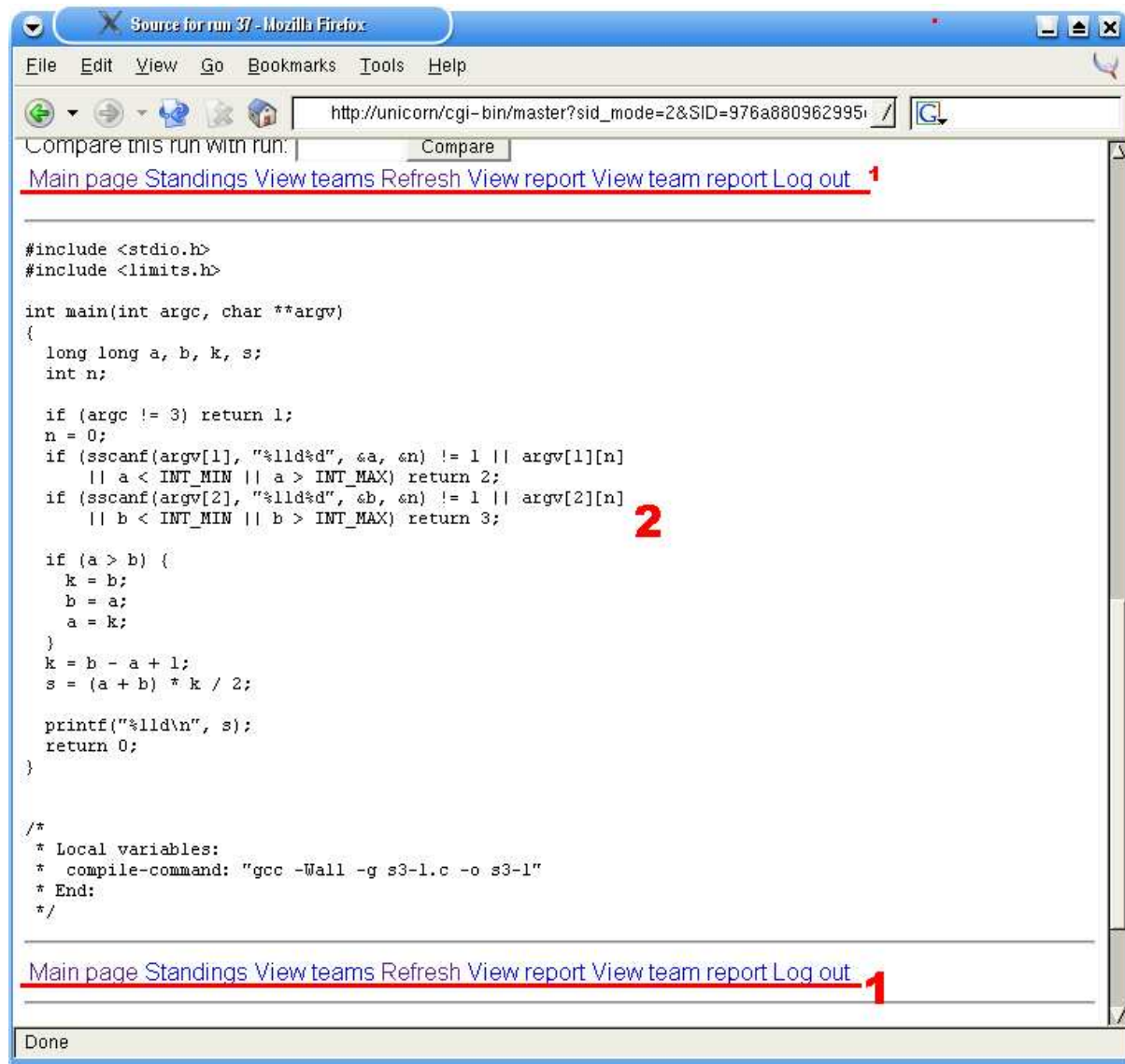


Рис. 4.13: Интерфейс просмотра исходного кода посылок

- “View team report” вызывает переход к странице просмотра протокола тестирования для текущей посылки, предназначенного для участника турнира. Данная ссылка отображается, только если глобальная конфигурационная переменная `team_enable_rep_view` конфигурационного файла `serve.cfg` установлена в `true`.
  - “Log out” приводит к выходу пользователя из программы `master` и завершению сессии работы.
- Здесь отображается текст посылки. Отображение текста возможно, только если конфигурационная переменная `binary` файла конфигурации турнира `serve.cfg` не установлена, то есть для языков программирования, исходные файлы которых представляют собой текстовые файлы.

#### 4.1.15 Просмотр протокола тестирования

На рис. 4.14 представлен вид окна браузера при отображении страницы протокола тестирования посылки. На копии экрана представлен вид судейского протокола тестирования, к просмотру которого можно перейти либо нажав на ссылку “View report” в таблице посылок на главной странице CGI-программ `serve` и `judge`, либо нажав на ссылку “View report” на странице редактирования посылки.

Если установлена конфигурационная переменная `team_enable_rep_view`, то привилегированным пользователям становится доступным и пользовательский протокол тестирования. К нему можно перейти, нажав на ссылку “View team report” на странице редактирования посылки.

Элементы отображения протокола тестирования перечислены ниже. Рассматривается протокол тестирования для турнира, проводимого по системе *KIROV*.

1. Ссылки быстрой навигации позволяют переключиться на другой экран программы `master`. Доступны следующие ссылки быстрой навигации:
  - “Main page” вызывает переход на главную страницу программы `master`.
  - “Standings” вызывает переход на страницу просмотра текущих результатов турнира.
  - “View teams” вызывает переход на страницу просмотра информации об участниках турнира.
  - “View source” вызывает переход на страницу редактирования посылки.
  - “Refresh” вызывает обновление текущей страницы.
  - “View team report” вызывает переход к странице просмотра протокола тестирования для текущей посылки, предназначенного для участника турнира. Данная ссылка отображается, только если глобальная конфигурационная переменная `team_enable_rep_view` конфигурационного файла `serve.cfg` установлена в `true`.
  - “Log out” приводит к выходу пользователя из программы `master` и завершению сессии работы.

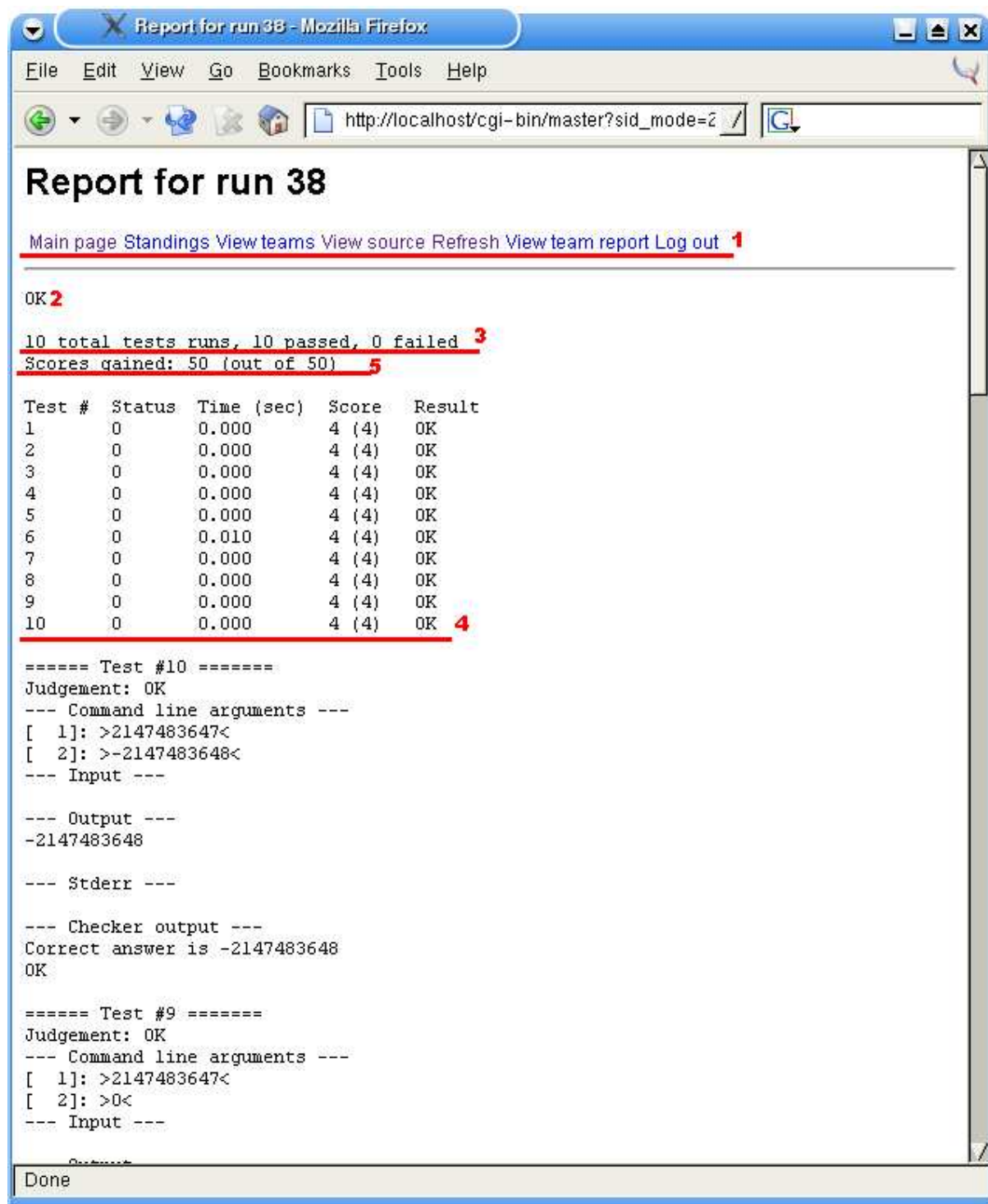


Рис. 4.14: Интерфейс просмотра протокола тестирования



2. Здесь отображается общий вердикт тестирования. Для турниров по системе *ACM* вердикт может быть “OK”, “Run-time error”, “Presentation error”, “Wrong answer”, “Check failed”. Для турниров по системе *KIROV* вердикт может быть одним из следующих: “OK”, “Partial solution”, “Check failed”. Для турниров по системе *OLYMPIAD* вердикт может быть одним из следующих: “OK”, “Partial solution”, “Check failed”, “Accepted for testing”.
3. Здесь отображается статистика по количеству пройденных тестов. Выводится сколько тестов по данной задаче находится в каталоге тестов (“total test runs”), сколько тестов было пройдено, и сколько тестов осталось не пройдено.
4. В данном разделе выводится подробные результаты тестирования по каждому тесту в наборе. Для каждого теста выводится:
  - Порядковый номер теста.
  - Код завершения программы на этом тесте. Ненулевые коды возврата интерпретируются как ошибка выполнения программы (“Run-time error”).
  - Время работы программы на данном тесте в секундах.
  - Балл, полученный за данный тест и полный балл за тест (печатается в скобках). Поскольку система **ejudge** не поддерживает выставление частичного балла за тест, решение может получить либо 0 баллов за тест, либо полный балл. Данный столбец не выводится в турнирах в режиме *ACM*, поскольку для таких турниров понятие балла за тест не определено.
  - Результат работы на каждом тесте. Результат работы может быть “OK”, “Run-time error”, “Time-limit exceeded”, “Presentation error”, “Wrong answer” и “Check failed”.
5. В поле “Score gained” отображается, сколько баллов было получено решением на тестировании и максимальное количество баллов. Количество баллов за тестирование определяется как сумма баллов, полученных за тесты. Количество баллов за тестирование, отображаемое на этой странице, выводятся без учёта штрафов за повторные попытки (см. [run\\_penalty](#)) и без учёта штрафов за календарное время сдачи.
6. Подробный протокол тестирования для каждого теста. В протоколе тестирования для теста отображаются:
  - Аргументы командной строки (Command line arguments), если конфигурационная переменная `use_info` секции описания соответствующей задачи в конфигурационном файле `serve.cfg` установлена в *true*, и если конфигурационная переменная `params` в файле описания теста `test.inf` не пуста.
  - Входной тестовый файл (Input). Если размер входного тестового файла превышает значение конфигурационной переменной `max_file_length`, вместо содержимого файла печатается сообщение “file is too long”. Если длина одной строки текста в этом файле превышает значение конфигурационной переменной `max_line_length`, вместо данной строки в протоколе печатается строка “Line is too long”.
  - Результат работы программы (Output). Если согласно описанию задачи результат работы должен быть напечатан на стандартный поток вывода (то есть если

установлена конфигурационная переменная `use_stdout` секции описания соответствующей задачи в конфигурационном файле `serve.cfg`.

На отображаемый файл также накладываются ограничения, определяемые глобальными конфигурационными переменными `max_file_length` и `max_line_length` файла конфигурации турнира `serve.cfg`.

- Выдача тестируемой программы на стандартный поток ошибок (Stderr). На отображаемый файл также накладываются ограничения, определяемые глобальными конфигурационными переменными `max_file_length` и `max_line_length` файла конфигурации турнира `serve.cfg`.
- Выдача проверяющей программы на данном тесте (Checker output). Для проверяющей программы и выдача в стандартный поток ошибок, и выдача в стандартный поток вывода сливаются в один поток вывода, отображаемый в данной секции.

#### 4.1.16 Интерфейс просмотра сообщений

На рис. 4.15 показан вид окна браузера при просмотре сообщения. Окно просмотра сообщения активизируется при нажатии на ссылку “View” в таблице сообщений.

1. Ссылки быстрой навигации позволяют переходить к окнам отображения другой информации программы **master**.
  - Ссылка “Main page” вызывает возврат на главную страницу программы **master**.
  - Ссылка “Standings” вызывает переход к привилегированной версии текущего положения участников турнира.
  - Ссылка “View teams” вызывает переход к странице списка участников турнира.
  - Ссылка “Log out” вызывает переход к заключительной странице программы **master** и завершение работы с программой.
2. В поле “Clar ID” отображается порядковый номер сообщения. Сообщения нумеруются от 0.
3. В поле “Flags” отображаются флаги, установленные для данного сообщения.
  - Флаг N означает, что сообщение поступило от участника турнира судьям, и ещё не было просмотрено с помощью программы **judge**. Просмотры сообщения программой **master** не сбрасывают флага N.
  - Флаг R означает, что сообщение поступило от участника турнира судьям, и было просмотрено с помощью программы **judge**.
  - Флаг A означает, что сообщение поступило от участника турнира судьям, и на него был дан ответ либо с помощью программы **master**, либо с помощью программы **judge**. Когда для сообщения устанавливается флаг A, одновременно устанавливается флаг R, и сбрасывается флаг N.
4. В поле “Time” отображается астрономическое время получения сообщения сервером турнира. Время отображается в стандартном формате системы **ejudge**.



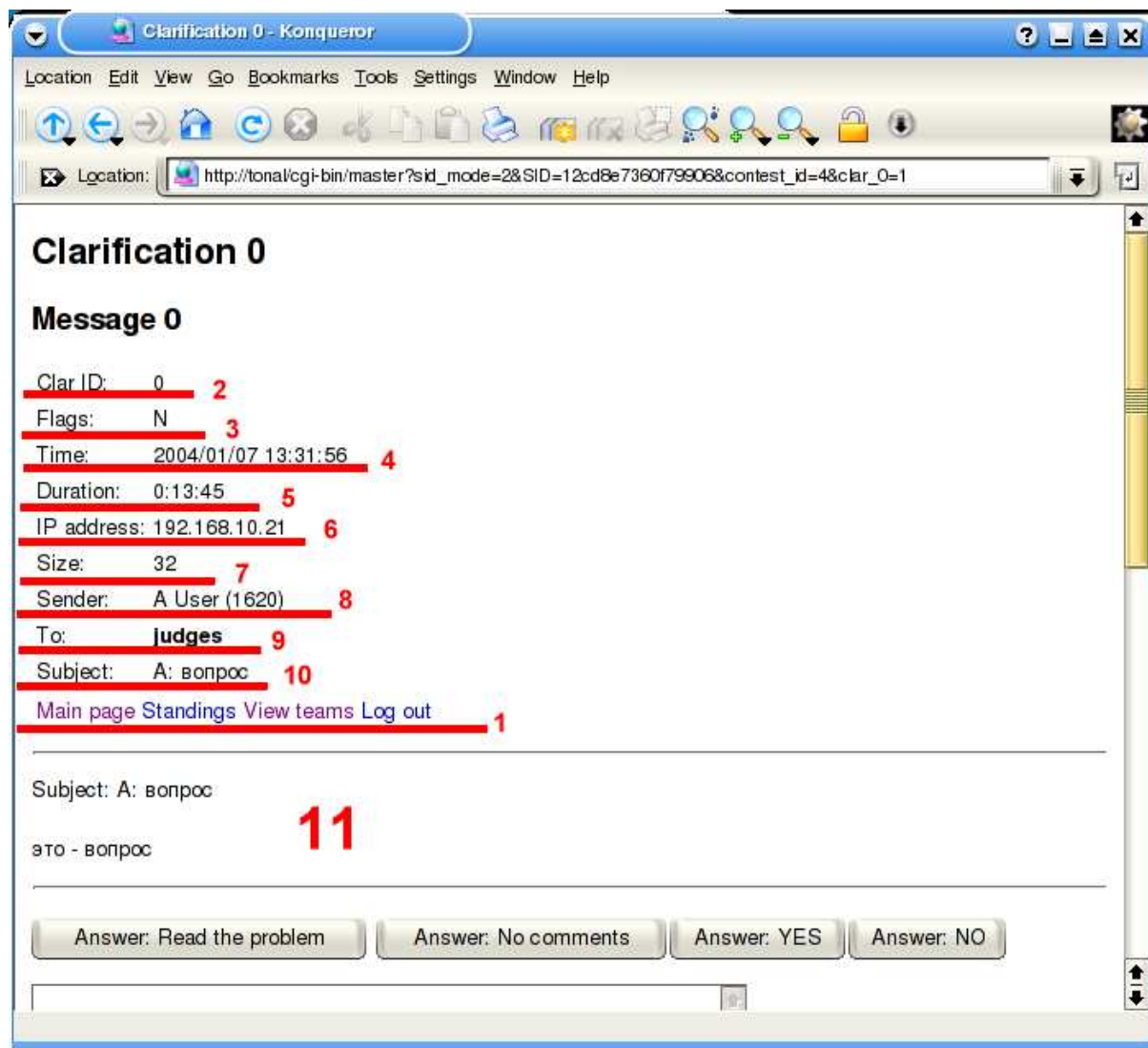


Рис. 4.15: Интерфейс просмотра сообщений

5. В поле “Duration” отображается время, прошедшее от момента начала турнира до получения сообщения сервером турнира. Продолжительность отображается в стандартном формате системы **ejudge**.
6. В поле “IP address” отображается IP-адрес браузера, с которого было послано сообщение.
7. В поле “Size” отображается размер сообщения. В размер сообщения входит и размер темы сообщения.
8. В поле “Sender” отображается отправитель сообщения. Если сообщение было отправлено с помощью привилегированной программы **master** или **judge**, в поле отправителя печатается “judges”. В противном случае печатается имя пользователя (name) или его регистрационное имя (login), если имя пользователя не установлено, а затем в скобках печатается его идентификатор пользователя. На нашем примере A User — это имя пользователя, а 1620 — его идентификатор.
9. В поле “To” отображается получатель сообщения. Если сообщение было отправлено непривилегированным пользователем с помощью CGI-программы **team**, в данном поле выводится “judges”, так как непривилегированный пользователь может отправлять сообщения только судьям. Если сообщение было отправлено привилегированным пользователем (т. е. с помощью программ **master** или **serve**) всем участникам турнира, в данном поле отображается “All”. Если сообщение было отправлено привилегированным пользователем некоторому конкретному пользователю, то печатается имя пользователя (name) или его регистрационное имя (login), если имя пользователя не установлено, а затем в скобках печатается его идентификатор пользователя.
10. В поле “Subject” отображается тема сообщения. Тема сообщения ограничивается 14 символами. Если тема сообщения длиннее 14 символов, то в конце ставится . . . , а полностью тему сообщения можно прочитать в окне отображения текста сообщения.
11. Здесь отображается полный текст сообщения. Первая строка текста — всегда строка Subject, в которой отображается полная тема сообщения. Строка Subject отделяется от тела сообщения одной пустой строкой.

#### 4.1.17 Интерфейс ответа на сообщение

На рис. 4.16 представлен вид окна браузера на нижнюю часть окна просмотра сообщения. В нижней части окна располагаются управляющие элементы для составления и отправки ответа на сообщения участников турнира.

При отправке ответа на любое сообщение в текст ответа всегда вставляется текст исходного сообщения, выделяемый с помощью знака > в начале строки (как при переписке по электронной почте). После цитирования текста исходного сообщения следует непосредственно текст ответа на него.

1. Ссылки быстрой навигации позволяют переходить к окнам отображения другой информации программы **master**.
  - Ссылка “Main page” вызывает возврат на главную страницу программы **master**.

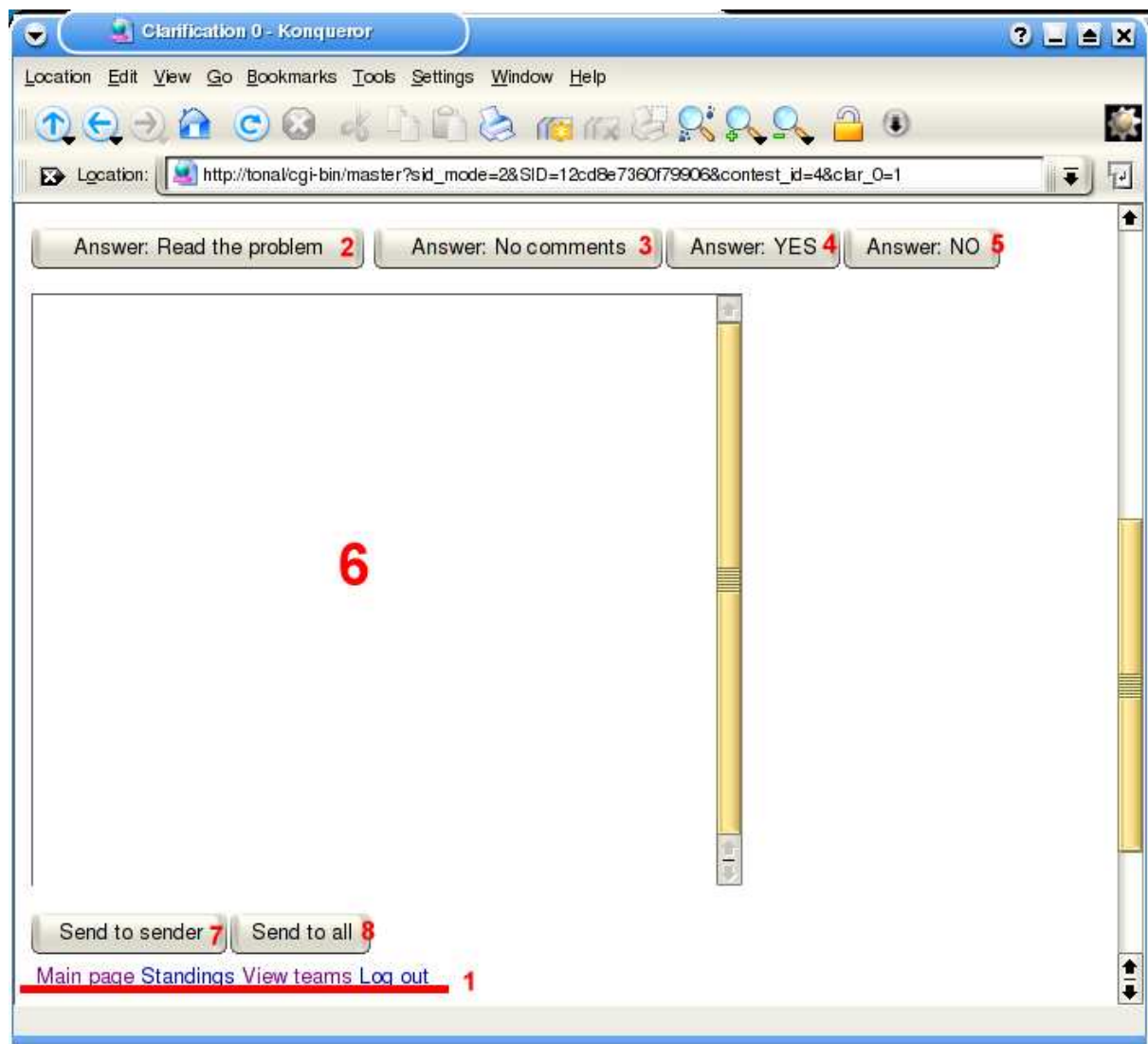


Рис. 4.16: Интерфейс ответа на сообщение

- Ссылка “Standings” вызывает переход к привилегированной версии текущего положения участников турнира.
  - Ссылка “View teams” вызывает переход к странице списка участников турнира.
  - Ссылка “Log out” вызывает переход к заключительной странице программы **master** и завершение работы с программой.
2. Кнопки быстрого ответа 2, 3, 4, 5 предназначены для выбора одного из фиксированных вариантов ответа и отправки ответа только участнику, от которого исходило данное сообщение. При использовании данных кнопок игнорируется текст, введённый в поле ввода текста ответа 6.
- Кнопка “Answer: Read the problem” генерирует текст ответа “Read the problem”. Данный ответ должен использоваться, когда, по мнению жюри, ответ на поставленный вопрос содержится в тексте условия.
3. Кнопка “Answer: No comments” генерирует текст ответа “No comments”. Данный ответ должен использоваться, когда, по мнению жюри, вопрос поставлен некорректно, или жюри не считает правильным отвечать на вопрос.
4. Кнопка “Answer: YES” генерирует текст ответа “YES”.
5. Кнопка “Answer: NO” генерирует текст ответа “NO”.
6. В данном поле вводится текст ответа на сообщение.
7. Кнопка “Send to sender” отправляет ответ, введённый в поле 6, только участнику турнира, задавшему данный вопрос.
8. Кнопка “Send to all” отправляет ответ, введённый в поле 6, всем участникам турнира.

# Глава 5

## Внутренние файлы

В настоящей главе описывается формат файлов, в которых хранится информация о турнире. Эти файлы обычно непосредственно недоступны как участникам, так и администраторам турнира, и доступ к ним осуществляется посредством программ системы **ejudge**.

### 5.1 Таблица посылок

Таблица посылок хранит всю информацию о посылках, кроме текста программы и протокола тестирования. По-умолчанию таблица посылок хранится в файле `run.log`, располагающемся в подкаталоге `var` каталога турнира. Имя файла таблицы посылок или полный путь к нему можно изменить с помощью глобальной конфигурационной переменной `run_log_file` конфигурационного файла турнира `serve.cfg`. Напомним, что каталог турнира задаётся с помощью конфигурационной переменной `root_dir` конфигурационного файла турнира `serve.cfg`, а также с помощью элемента `root_dir` конфигурационного файла турнира `contest.xml`.

Таблица посылок хранится в бинарном формате. В начале файла находится заголовок файла, который занимает 64 байта. Затем идёт произвольное количество записей фиксированной длины. Каждая запись также имеет размер 64 байта. Обратите внимание, что дисциплина хранения исходных файлов решений и файлов протоколов тестирования, используемая в настоящий момент, ограничивает максимальное количество записей в таблице не более чем 100000. Возможно, это ограничение будет в дальнейшем пересмотрено.

#### 5.1.1 Заголовок таблицы

Формат заголовка таблицы описывается следующей структурой данных на языке Си.

```
struct run_header
{
    int      version;
    time_t   start_time;
    time_t   sched_time;
    time_t   duration;
    time_t   stop_time;
    unsigned char pad[44];
};
```

Здесь предполагается, что тип `int` имеет размер 4 байта (32 бита), тип `time_t` является синонимом типа `long`, который также имеет размер 4 байта (32 бита). Тип `unsigned char` имеет размер 1 байт (8 битов). Для платформ с другими размерами примитивных типов определение структуры заголовка должно быть соответствующим образом изменено. Целые значения хранятся в том порядке байтов, который принят на данной платформе. Например, на `Linux-ix86` целые значения хранятся в порядке байтов “Little-endian”.

Поле `version` хранит номер версии формата базы посылок. Текущая версия имеет номер 1. Номер версии будет увеличиваться каждый раз, когда формат базы посылок меняется несовместимым с предыдущими версиями образом.

Поле `start_time` хранит фактическое время начала турнира, если турнир уже начался. Если турнир ещё не начался, в данном поле хранится значение 0. Время хранится во внутреннем формате `Unix-систем`, то есть как количество секунд, прошедшее от 1 января 1970 года.

Поле `sched_time` хранит назначенное время начала турнира. Для уже начавшегося турнира (то есть для турнира, у которого `start_time` не равно 0) значение этого поля не имеет значения. Если значение этого поля равно 0, то для данного турнира никакое время старта не было назначено. Назначить время старта турнира можно с помощью интерфейса администратора турнира, предоставляемого CGI-программой `master`. В момент первого запуска сервера турнира, когда файл базы посылок создаётся, в данное поле записывается значение 0.

Поле `duration` хранит установленную продолжительность турнира в секундах. Для турниров неограниченной продолжительности данное поле содержит значение 0. Изменить назначенную продолжительность турнира можно с помощью интерфейса администратора турнира, предоставляемого CGI-программой `master`. В момент первого запуска сервера турнира для данного турнира при создании файла базы посылок в данное поле копируется значение глобальной конфигурационной переменной `contest_time` конфигурационного файла турнира `serve.cfg`.

Поле `stop_time` хранит фактическое время окончания турнира. Если турнир ещё не окончен, данное поле хранит значение 0. Турнир заканчивается автоматически, когда истекает его продолжительность, либо турнир в любой момент может быть закончен по команде администратора с помощью интерфейса, предоставляемого CGI-программой `master`.

Поле `pad` используется для дополнения структуры до размера 64 байта.

### 5.1.2 Запись о посылке в таблице

Формат одной записи описывается следующей структурой данных на языке Си. Как было сказано выше, размер такой структуры на архитектуре `Linux-ix86` равен 64 байтам.

```
struct run_entry
{
    int            submission;
    time_t         timestamp;
    size_t         size;
    unsigned long  ip;
    unsigned long  sha1[5];
    int            team;
    int            problem;
    int            score;
    signed char    locale_id;
```

```

unsigned char language;
unsigned char status;
signed char test;
unsigned char is_imported;
unsigned char variant;
unsigned char is_hidden;
unsigned char is_readonly;
unsigned char pages;
unsigned char pad[3];
long nsec;
};

```

Здесь используются предположения о размере и размещении целых типов, справедливые для архитектуры Linux-ix86. Далее даётся подробное описание назначения каждого из полей структуры.

Поле `submission` хранит номер посылки. Каждая посылка нумеруется от 0 и до некоторого максимального значения. Дисциплина хранения файлов с исходным текстом программы и протоколов тестирования посылки в настоящее время такова, что максимальный номер посылки не может превышать 100000. Номер посылки должен быть проставлен для всех записей в базе посылок, даже для записей со статусом *EMPTY*, при этом номер посылки должен совпадать с расположением структуры соответствующей посылки в файле. Так, непосредственно за заголовком файла должна идти посылка с номером 0, затем посылка с номером 1 и т. д. Пропуски в последовательной нумерации посылок не допускаются. Посылки должны размещаться в файле посылок в строго возрастающем порядке.

Поле `timestamp` содержит время приёма данной посылки сервером турнира `serve`. Время хранится во внутреннем формате Unix-систем, то есть как число секунд, прошедших с 1 января 1970 года. Все посылки в файле посылок должны быть упорядочены в порядке неубывания их времени приёма за исключением записи со статусом *EMPTY*. Время приёма посылки может быть меньше времени начала турнира или больше времени конца турнира только для посылок, у которых поле `is_hidden` установлено в *true*. Для записи в базе посылок со статусом *EMPTY* значение данного поля должно быть равным 0.

Поле `nsec` содержит наносекундную компоненту времени приёма данной посылки сервером турнира. Данное поле может принимать значения в интервале от 0 до 999999999. Для каждой посылки в турнире полное время её приёма (то есть время, включающее как секундную, так и наносекундную компоненты) уникально. Посылки в базе посылок всегда упорядочиваются по полному времени приёма посылки. Данное поле введено в `ejudge` версии 2.1.27, для турниров, обслуживаемых предыдущей версией системы, наносекундная компонента времени всегда равна нулю. Формат журнала посылки совместим снизу вверх и сверху вниз, то есть для `ejudge` младших версий поле `nsec` будет проигнорировано, а для `ejudge` старших версий значение поля `nsec` по умолчанию будет установлено в 0.

Поле `size` содержит размер исходного файла программы в байтах. Протокол взаимодействия CGI-программы и сервера турнира накладывает ограничения на максимальный размер программы. Кроме того, CGI-программы `master`, `judge`, `team` не принимают данные от веб-браузера по методу POST, если размер данных превышает ? кб. Кроме того, максимальный размер принимаемых на проверку программ может быть установлен в глобальной конфигурационной переменной `max_run_size` конфигурационного файла сервера турнира `serve.cfg`. Значение этой переменной по умолчанию устанавливается в 65535 байтов, но не может превышать ограничения на приём данных по методу POST и ограничения протокола. Для записей в базе посылок со статусом *EMPTY*, *VSTART* или *VSTOP* поле `size` должно



содержать 0.

Обратите внимание, что текущая версия системы **ejudge** передаёт программу как одну строку, то есть как последовательность байт, в которой нулевой байт означает конец строки. Поэтому в настоящее время не поддерживаются нулевые символы в тексте программы, а также сдача программ в каком-либо бинарном (или архивном) формате.

Поле `ip` содержит IP-адрес (IPv4) хоста, с которого с помощью веб-браузера было отослано решение. Старший байт поля IP-адреса в данной структуре содержит первый октет IP-адреса в точечной нотации. Таким образом IP-адрес хранится в локальном (а не сетевом) порядке байт в слове. Для записи в базе посылок со статусом *EMPTY* данное поле должно содержать значение 0.

Поле `sha1` содержит контрольную сумму исходного текста программы, вычисляемую по алгоритму SHA1. Этот алгоритм любой последовательности байт ставит в соответствие 160-битное число, при этом вероятность коллизии крайне мала. Если два файла имеют одинаковую контрольную сумму SHA1, они с вероятностью близкой к единице идентичны. Для записей в базе посылок со статусом *EMPTY*, *VSTART* или *VSTOP* данное поле должно содержать нулевую контрольную сумму.

Контрольная сумма используется для того, чтобы автоматически выявлять дубликаты посылок. Таковыми считаются посылки, у которых совпадают IP-адрес, размер, контрольная сумма, идентификатор участника, идентификатор задачи и идентификатор языка программирования. Дубликаты посылок не проверяются и им немедленно присваивается статус “Ignored”. Дубликаты посылок могут возникать, если пользователь после отправки решения на проверку, не дождавшись подтверждения о её получении, нажимает кнопку браузера “Refresh”.

Поле `team` хранит идентификатор пользователя, от имени которого выполнена данная посылка. Идентификатор пользователя представляет собой положительное целое число (идентификатор пользователя 0 является недопустимым). Текущая версия системы **ejudge** накладывает верхнее ограничение на идентификатор пользователя, равное 100000. Для записи в базе посылок со статусом *EMPTY* данное поле должно содержать значение 0.

Поле `problem` хранит идентификатор задачи. Идентификатор задачи представляет собой целое число, большее нуля (0 как идентификатор задачи недопустим). Система **ejudge** накладывает верхнее ограничение на идентификатор задачи, равное максимальному числу задач, которые могут быть определены в одном турнире. В настоящее время это число равно 100 задачам. Идентификатор задачи в записи базы посылок должен соответствовать идентификатору какой-либо задачи в конфигурационном файле турнира `serve.cfg`. Идентификатор задачи можно установить явно с помощью конфигурационной переменной `id` раздела описания задачи этого файла.

Поле `score` хранит количество баллов, полученное при тестировании данной отправки, без учёта штрафов за повторные отправки. Это поле используется только в режимах турнира *KIROV* и *OLYMPIAD*. Количество баллов представляет собой неотрицательное целое число. Значение  $-1$  данного поля задаёт неопределённое значение, которое отображается в журнале посылок как N/A. Максимальное число баллов в текущей версии установлено в 100000.

Поле `locale_id` хранит идентификатор языкового окружения, установленный в клиенте **team** в момент отправки отправки. Идентификатор языкового окружения представляет собой неотрицательное целое число от 0 до 127. Значение  $-1$  поля `locale_id` задаёт «неопределённое» значение. В этом случае программы `serve` и `run` используют значения по умолчанию (в настоящее время 0). Идентификатор языкового окружения используется программой `run` при генерации пользовательского протокола тестирования.

В настоящее время система **ejudge** поддерживает два языковых окружения: английское



(идентификатор 0) и русское (идентификатор 1). В обоих случаях используется кодировка `koi8-r`.

Поле `language` хранит идентификатор языка программирования, на котором написана посылаемая программа. Идентификатор языка программирования представляет собой целое число в интервале от 0 до 255. Идентификатор языка программирования в записи базы посылок должен соответствовать идентификатору какого-либо языка программирования в конфигурационном файле турнира `serve.cfg`. Идентификатор языка программирования для некоторого языка можно установить явно с помощью конфигурационной переменной `id` раздела описания языка программирования этого файла.

Поле `status` хранит статус посылки. Статус посылки подробно рассматривается в таблице 5.1.

Поле `test` хранит либо минимальный номер теста, на котором программа дала неверный ответ (для турниров по системе *ACM*), либо количество тестов, на которых программа дала правильный ответ (для турниров по системе *KIROV* или *OLYMPIAD*).

Для турниров по системе *ACM* если значение данного поля  $\leq 0$ , то минимальный номер теста не определен (в случаях, когда статус посылки допускает это, например, при статусе “Compilation error”). Для турниров по системам *KIROV* или *OLYMPIAD* значение данного поля, не большее 0 также означает, что количество тестов, на которых программа дала правильный ответ, не определено. Если для турниров по этим системам поле `test` хранит положительное значение, то оно на 1 больше количества успешно пройденных тестов. Например, если решение дало правильный ответ на 3 тестах, в базе посылок в поле `test` хранится значение 4.

Поле `is_imported` хранит признак проимпортированной посылки. Поле может принимать два значения: 0 и 1, соответствующие булевским значениям *false* и *true*. У посылки устанавливается этот флаг при импорте в базу посылок внешнего XML-файла с посылками, если импортируемой посылке не соответствует никакая локальная посылка. В текущей версии системы **ejudge** при импорте базы посылок не импортируется исходный код программы посылки, а также протоколы тестирования. Поэтому при работе с посылкой, у которой поле `is_imported` установлено в *true* предполагается, что ни исходного файла, ни файлов протоколов тестирования в архивах турнира не существует. Как следствие, такие посылки не могут быть пересужены с помощью кнопок “Rejudge” интерфейса администратора турнира. При необходимости нужный статус посылки должен быть выставлен вручную. Обратите внимание, что результаты редактирования проимпортированной посылки могут потеряться при очередном импорте, если эта посылка в импортируемом XML-файле помечена как «авторитетная».

Поле `variant` хранит номер варианта задачи. Для невариантных задач данное поле не используется. Для вариантных задач данное поле может иметь значение 0, что означает, что должен использоваться вариант согласно таблице вариантов (см. глобальную конфигурационную переменную `variant_map_file`), либо данное поле может хранить номер варианта явно. Явный номер варианта задачи может выставляться только администратором турнира при редактировании записи, либо выбираться при отправке посылки из-под привилегированной CGI-программы **master** или **judge** пользователем с соответствующими привилегиями.

Поле `is_hidden` хранит флаг скрытой посылки. Поле может принимать два значения: 0 и 1, соответствующие булевским значениям *false* и *true*. Флаг скрытой посылки ставится всегда при отправке посылки из-под привилегированных CGI-программ **master** или **judge**. Кроме этого флаг скрытой посылки может быть установлен вручную администратором турнира.

Когда флаг скрытой посылки установлен в *true*, такая посылка:

- Не учитывается при вычислении штрафов за попытки, текущих результатов турнира и не отображается в таблице текущих результатов и публичном журнале посылок (если таковой ведётся).
- Не экспортируется в XML-формате.
- Может иметь время получения до момента начала турнира и после момента окончания турнира.

Поле `is_readonly` хранит флаг нередактируемой посылки. Поле может принимать два значения: 0 и 1, соответствующие булевским значениям *false* и *true*. Флаг нередактируемой посылки может быть установлен вручную администратором турнира.

Если флаг нередактируемой посылки установлен в *true*, данная посылка никогда не пере-суживается с помощью кнопок “Rejudge” интерфейса администратора турнира. Администратор турнира не может изменить ни одно поле посылки, за исключением поля `is_readonly`.

Назначение данного флага — предотвратить случайное изменение полей посылки (например, с помощью кнопки “Rejudge”, если статус посылки был установлен вручную), а не запретить каким-либо пользователям редактировать поля базы посылок.

Поле `pages` хранит количество страниц бумаги, использованных при распечатке программы непривилегированным пользователем. Если значение поля равно 0, программа не была распечатана. Значение поля устанавливается после распечатки исходного кода программы в данной посылке. Счётчик страниц включает в себя обязательную титульную страницу, поэтому даже самая маленькая программа потребует две страницы.

По умолчанию распечатка программ непривилегированными пользователями запрещена. Чтобы разрешить распечатку необходимо установить в *true* глобальную конфигурационную переменную `enable_printing` конфигурационного файла описания турнира `serve.cfg`.

### 5.1.3 Статус посылки

В таблице 5.1 перечислены все возможные статусы записей в базе посылок. Все перечисленные статусы подробно комментируются ниже.

В столбце «Типы турниров» перечислены типы турниров, в которых посылка может иметь указанный статус. Букве *A* соответствует тип турнира *ACM*, букве *K* — *KIROV*, а букве *O* — *OLYMPIAD*.

- **OK**. Посылка с таким статусом соответствует полностью засчитанному решению. Решение засчитывается полностью, если оно даёт правильный ответ на всех тестах. Кроме того, статус OK может быть установлен вручную администратором системы.

В турнире по системе *ACM* участники ранжируются по количеству полностью засчитанных решений. Полностью засчитанное решение получает штраф равный количеству минут, прошедший от момента начала турнира до момента получения этого решения плюс по 20 штрафных минут за каждую предыдущую попытку сдачи этой задачи. Все послылки некоторого участника по некоторой задаче посланные после успешной посылки игнорируются.

В турнире по системе *KIROV* полностью засчитанное решение получает либо полный балл, указанный в конфигурационной переменной `full_score` секции описания задачи `problem` конфигурационного файла описания турнира `serve.cfg`, либо количество баллов, указанное в поле `score`, если конфигурационная переменная

Обозначение	Чи- сло- вое зна- чение	Вы- раже- ние филь- тра	Типы тур- ниров	Английский текст	Русский текст
<b>Обычные состояния</b>					
OK	0	OK	<i>AKO</i>	OK	OK
COMPILE_ERR	1	CE	<i>AKO</i>	Compilation error	Ошибка компиляции
RUN_TIME_ERR	2	RT	<i>A</i>	Run-time error	Ошибка выполнения
TIME_LIMIT_ERR	3	TL	<i>A</i>	Time-limit exceeded	Превышено максимальное время работы
PRESENTATION_ERR	4	PE	<i>A</i>	Presentation error	Неправильный формат вы- вода
WRONG_ANSWER_ERR	5	WA	<i>A</i>	Wrong answer	Неправильный ответ
CHECK_FAILED	6	CF	<i>AKO</i>	Check failed	Проверка завершилась неудачно
PARTIAL	7	PT	<i>KO</i>	Partial solution	Неполное решение
ACCEPTED	8	AC	<i>AKO</i>	Accepted for testing	Принято на проверку
IGNORED	9	IG	<i>AKO</i>	Ignored	Проигнорирована
<b>Специальные записи</b>					
VIRTUAL_START	20	VS	<i>A</i>	Virtual start	Начало виртуального тур- нира
VIRTUAL_STOP	21	VT	<i>A</i>	Virtual stop	Окончание виртуального турнира
EMPTY	22	EM	<i>AKO</i>	EMPTY	ПУСТО
<b>Промежуточные состояния</b>					
RUNNING	96	RU	<i>AKO</i>	Running...	Выполняется...
COMPILED	97	CD	<i>AKO</i>	Compiled	Скомпилировано
COMPILING	98	CG	<i>AKO</i>	Compiling...	Компилируется...
AVAILABLE	99	AV	<i>AKO</i>	Available	Доступна

Таблица 5.1: Допустимые статусы посылок

`variable_full_score` секции описания задачи установлена в значение `true`. Далее из этого количества баллов вычитается штраф за предыдущие попытки (как успешные, так и неуспешные). Штраф за попытку устанавливается в конфигурационной переменной `run_penalty`. Если получающееся число оказывается меньше нуля, берётся значение 0. Окончательное число баллов, полученное участником за задачу берётся как максимум по всем попыткам этого пользователя сдать данную задачу.

В турнир по системе *OLYMPIAD* отличается от турнира по системе *KIROV* тем, что штраф за предыдущие попытки не начисляется, а количество баллов за задачу берётся по последней попытке сдать эту задачу.

- **COMPILE\_ERR.** Если у послышки установлен такой статус, при компиляции программы возникла ошибка компиляции. Причины возникновения ошибки компиляции зависят от используемого компилятора. Судейский протокол тестирования для посылок с таким статусом содержит полный вывод (на стандартный поток вывода и стандартный поток ошибок) компилятора. Если глобальная конфигурационная переменная `team_enable_rep_view` или конфигурационная переменная раздела описания задачи `team_enable_rep_view` файла конфигурации турнира `serve.cfg` установлена в `true`, то диагностика компилятора становится доступной участнику турнира.

Кроме того, если установлена глобальная конфигурационная переменная `team_enable_ce_view` или конфигурационная переменная раздела описания задачи `team_enable_ce_view` файла конфигурации турнира `serve.cfg` установлена в `true`, диагностика компилятора в случае ошибки компиляции предоставляется участнику вне зависимости от значения конфигурационной переменной `team_enable_rep_view`.

Если установлена глобальная конфигурационная переменная `ignore_compile_errors` файла конфигурации турнира `serve.cfg`, то послышки, имеющие данный статус, не учитываются при вычислении текущих результатов турнира, штрафов за предыдущие послышки участника, набранного количества баллов и т. д.

- **RUN\_TIME\_ERR.** Данный статус специфичен для турниров по системе *ACM* и устанавливается, когда на некотором тесте тестируемая программа завершилась аварийно. Для турниров по системам *OLYMPIAD* и *KIROV* аварийное завершение тестируемой программы на каком-либо тесте приведёт к тому, что статус послышки будет установлен в `PARTIAL` или `CHECK_FAILED` в зависимости от результата тестирования на других тестах.

Тестируемая программа считается завершившейся аварийно, если процесс был завершён из-за получения им сигнала, который по умолчанию вызывает завершение процесса (например, `SIGSEGV`), или если процесс завершился с ненулевым кодом возврата.

Если включён режим передачи кода завершения процесса через файл (см. параметр `errorcode_file` секции тестировщика конфигурационного файла турнира `serve.cfg`, то проверяется только код возврата, записанный в этом файле, а статус завершения процесса игнорируется.

При аварийном завершении тестируемой программы её вывод не проверяется, хотя и вставляется в судейский протокол тестирования.

- **TIME\_LIMIT\_ERR.** Данный статус специфичен для турниров по системе *ACM* и устанавливается, когда на некотором тесте тестируемая программа превысила максималь-

ное время работы. Для турниров по системам *OLYMPIAD* и *KIROV* превышение максимального времени работы на каком-либо тесте приведёт к тому, что статус послышки будет установлен в `PARTIAL` или `CHECK_FAILED` в зависимости от результата тестирования на других тестах.

Тестируемая программа считается превысившей максимальное время работы, если виртуальное время работы процесса (пользовательская составляющая) превысило ограничение, установленное в конфигурационной переменной `time_limit` конфигурационного файла турнира `serve.cfg`, или если реальное (астрономическое) время работы процесса превысило ограничение, установленное в конфигурационной переменной `real_time_limit` того же файла. По умолчанию лимит астрономического времени установлен в 30 секунд. Рекомендуется устанавливать этот лимит так, чтобы он в 2—3 раза превосходил значение конфигурационной переменной `time_limit`.

Необходимость ограничения астрономического времени обоснована тем, что в режиме «сна» (например, вызвав функцию `pause`) процесс не потребляет виртуальное процессорное время и, как следствие, такой процесс может заблокировать проверяющую систему. С другой стороны, нельзя ограничивать время тестирования только с использованием астрономического времени, так как тестирующий компьютер может быть загружен неравномерно, могут выполняться другие процессы и т. д.

При превышении максимального времени работы тестируемой программы её вывод не проверяется, хотя и вставляется в судейский протокол тестирования.

- **PRESENTATION\_ERR.** Данный статус специфичен для турниров по системе *ACM* и устанавливается, если проверяющая программа по тем или иным причинам не может проанализировать вывод тестируемой программы на некотором тесте. Для турниров по системам *OLYMPIAD* и *KIROV* та же самая ошибка на каком-либо тесте приведёт к тому, что статус послышки будет установлен в `PARTIAL` или `CHECK_FAILED` в зависимости от результата тестирования на других тестах.

Если тестируемая программа завершилась корректно (с кодом возврата 0) и уложилась в отведённое время, запускается проверяющая программа, задача которой — сравнить результат работы тестируемой программы, сохранённый в файле, и правильный ответ для этого теста. Имя проверяющей программы задаётся в конфигурационной переменной `check_cmd` секции описания задачи конфигурационного файла турнира `serve.cfg`.

Проверяющая программа пишется специально для каждой задачи, хотя возможно повторное использование проверяющих программ в простых случаях, например, когда сравниваются два числа. Проверяющая программа на основании входного файла с тестом, результата работы программы на этом тесте и, возможно, файла с правильным ответом выдаёт вердикт либо `OK`, либо `PRESENTATION_ERR`, либо `WRONG_ANSWER_ERR`, либо `CHECK_FAILED`.

Традиционно, `PRESENTATION_ERR` выдаётся, когда проверяющая программа не может разобрать файл с результатом работы программы. Например, вместо целого числа в файле записана строка и т. д. Следует заметить, что граница между ошибками типа `PRESENTATION_ERR` и `WRONG_ANSWER_ERR` не всегда может быть определена точно.

- **WRONG\_ANSWER\_ERR.** Данный статус специфичен для турниров по системе *ACM* и устанавливается, когда проверяющая программа выносит соответствующий вердикт.

Для турниров по системам *OLYMPIAD* и *KIROV* та же самая ошибка на каком-либо тесте приведёт к тому, что статус послышки будет установлен в `PARTIAL` или `CHECK_FAILED` в зависимости от результата тестирования на других тестах.

Традиционно, `WRONG_ANSWER_ERR` выдаётся, если файл с результатом работы соответствует спецификации вывода, оговоренной в условии задачи, но конкретные значения не совпадают с правильными, например, правильный ответ — это целое число 4, а программа выдала в качестве ответа целое число 3. Следует заметить, что граница между ошибками типа `PRESENTATION_ERR` и `WRONG_ANSWER_ERR` не всегда может быть определена точно.

- **CHECK\_FAILED.** Данный статус устанавливается, если посылка не может быть проверена из-за сбоя в проверяющей системе. Для турнира по системе *ACM* этот статус устанавливается, если при тестировании программы на некотором тесте была получена ошибка `CHECK_FAILED`, а тестирование на всех предыдущих тестах завершилась успешно. Для турнира по системам *OLYMPIAD* или *KIROV* этот статус устанавливается, если хотя бы на одном тесте была получена ошибка `CHECK_FAILED`.

Ошибка `CHECK_FAILED` может возникнуть и до начала тестирования на этапе компиляции программы по следующим причинам:

- возникла ошибка копирования текста программы в разделяемые каталоги программы `compile`, например, из-за нехватки дискового пространства;
- возникла ошибка копирования файлов в программе `compile`;
- программа компиляции не может быть запущена (например, не найден файл программы);
- процесс компиляции завершился из-за получения сигнала, обрабатываемого по умолчанию;
- процесс компиляции не завершился за астрономическое время, установленное в конфигурационной переменной `compile_real_time_limit` секции описания языка программирования конфигурационного файла `serve.cfg`;
- процесс компиляции вернул нулевой статус завершения, но файл с исполняемым кодом не был создан.

На этапе тестирования программы ошибка `CHECK_FAILED` может возникнуть по следующим причинам:

- возникла ошибка копирования текста программы в разделяемые каталоги программы `run`, например, из-за нехватки дискового пространства;
- возникла ошибка копирования файлов в программе `run`;
- тестируемая программа не может быть запущена на выполнение;
- проверяющая программа не может быть запущена на выполнение;
- проверяющая программа была завершена из-за получения сигнала, обрабатываемого по умолчанию, или вернула код завершения, не равный кодам `OK`, `PRESENTATION_ERR` или `WRONG_ANSWER_ERR`;
- проверяющая программа не завершилась за астрономическое время, установленное в конфигурационной переменной `checker_real_time_limit` секции описания задачи конфигурационного файла `serve.cfg`.



В некоторых случаях судейский протокол тестирования будет содержать подробную диагностику о причине возникновения ошибки.

Посылка со статусом `CHECK_FAILED` не учитывается при вычислении текущего положения участников турнира, расчёте штрафных баллов и т. д.

- **PARTIAL**. Данный статус посылки специфичен для турниров типа *OLYMPIAD* и *KIROV*. Он устанавливается, когда ни на одном тесте не было получено ошибки `CHECK_FAILED`, но в то же время не на всех тестах был получен результат `OK`. Статус посылки `PARTIAL` — обычный статус для неполных решений в турнирах данных типов.
- **ACCEPTED**. Данный статус устанавливается у посылки в случаях, когда автоматическое тестирование не производится. Программа участника принимается и сохраняется в архиве исходных текстов, но тестирование программы будет проводиться позднее.

Посылка получает данный статус автоматически в следующих случаях.

- Турнир проводится по системе *OLYMPIAD* и находится в режиме сбора решений. В режиме сбора решений каждое поступающее решение проверяется на нескольких начальных тестах из набора тестов к задаче (количество тестов задаётся с помощью конфигурационной переменной `tests_to_accept` секции описания задачи конфигурационного файла описания турнира `serve.cfg`). Если поступившее решение проходит все тесты, соответствующей посылке присваивается статус `ACCEPTED`. В дальнейшем в режиме окончательного тестирования такие посылки будут протестированы уже на всех тестах.
- Турнир проводится по любой системе, но для некоторой задачи значение конфигурационной переменной `disable_auto_testing` конфигурационного файла описания турнира `serve.cfg` равно `true`. В этом случае при получении решения его немедленное тестирование не производится, но при выборе команды “Rejudge” любого вида, затрагивающей данную посылку, эта посылка будет полностью протестирована.
- Турнир проводится по любой системе, но для некоторой задачи значение конфигурационной переменной `disable_testing` раздела описания задачи конфигурационного файла описания турнира `serve.cfg` равно `true`. В этом случае тестирование решения средствами системы `ejudge` полностью отключено.
- Турнир проводится по любой системе, но находится в режиме приостановки автоматического тестирования поступающих решений. Турнир может быть переведён в этот режим командой “Suspend testing” администратора турнира (см. раздел 4.1.4). Все поступающие посылки получают статус `ACCEPTED` и могут быть впоследствии пересужены с помощью команды “Rejudge”.

Посылка со статусом `ACCEPTED` не учитывается при вычислении текущего положения участников турнира, расчёте штрафных баллов и т. д.

- **IGNORED**. Посылка с таким статусом не учитывается при вычислении текущего положения участников турнира, расчёте штрафных баллов и т. д.

Статус `IGNORED` присваивается посылке автоматически, если глобальная конфигурационная переменная `ignore_duplicated_runs` конфигурационного файла описания турнира `serve.cfg` равна `true` и в базе посылок уже существует посылка, у

которой значения полей `size`, `ip`, `sha1`, `team`, `problem` и `language` равны значениям этих полей текущей посылки. Такая проверка полезна, чтобы избежать дублирующих посылок (и начисления штрафных баллов) в случаях, когда пользователь системы слишком часто нажимает кнопку “Refresh” браузера при медленном канале.

При отправке решений из-под привилегированных программ `master` и `judge` проверка дубликатов никогда не выполняется.

Как и в случае других статусов посылок данный статус может быть установлен вручную при редактировании информации о посылке.

- **VIRTUAL\_START**. Такой статус имеет специальная запись в базе посылок, которая не соответствует никакой посылке участника. Она добавляется в базу посылок в момент начала участником своего виртуального турнира (см. глобальную конфигурационную переменную `virtual` конфигурационного файла описания турнира `serve.cfg`).

Запись такого типа может быть добавлена в базу посылок только при старте участником виртуального турнира и не может быть установлена при редактировании полей посылок средствами CGI-программы `master`. Запись может быть удалена только в том случае, если в базе посылок не осталось посылок и записи типа `VIRTUAL_STOP`, принадлежащих этому участнику.

- **VIRTUAL\_STOP**. Такой статус имеет специальная запись в базе посылок, которая не соответствует никакой посылке участника. Она добавляется в базу посылок в момент окончания участником своего виртуального турнира с помощью команды “Stop contest”. Если виртуальный турнир участника завершается автоматически по истечению времени турнира, никакая запись в базу посылок не добавляется.

Запись такого типа может быть добавлена в базу посылок только участником при принудительном завершении им своего виртуального турнира. Такой статус записи не может быть установлена при редактировании полей посылок средствами CGI-программы `master`.

- **EMPTY**. Запись такого типа никогда не добавляется в базу посылок автоматически. Статус `EMPTY` не может быть установлена при редактировании полей посылок средствами CGI-программы `master`.

Запись типа `EMPTY` возникает после очистки записи базы посылок с помощью команды “Clear entry” CGI-программы `master`. Поскольку номер посылки используется как первичный ключ для обращения к архиву исходных текстов и протоколов тестирования, операция сдвига записей в базе посылок на место удаляемых посылок является достаточно дорогой и поэтому не выполняется автоматически при каждом удалении записи. Чтобы выполнить сдвиг записей на место удалённых записей необходимо явно выполнить команду “Squeeze runs” CGI-программы `master`.

- **RUNNING**. Такой статус посылки устанавливается после копирования файла со служебной информацией и файла с исполняемым файлом в каталог обмена программы `run`. При получении результата тестирования от программы `run` статус посылки будет изменён на результат тестирования.

Если сервер компиляции `run` не запущен, файл со служебной информацией имеет неправильный формат, или сервер тестирования не может записать файл с результатом тестирования в каталог обмена, посылка может оставаться в состоянии `RUNNING` неограниченно долго. В этом случае необходимо вручную пересудить эту посылку.



- **COMPILED.** Такой статус послылки устанавливается после чтения файлов результата компиляции и до копирования исполняемого файла в каталог обмена с программой `run`, если компиляция завершилась успешно. После того, как все необходимые данные будут скопированы в каталог обмена, статус послылки меняется на `RUNNING`.
- **COMPILING.** Такой статус послылки устанавливается после копирования текста программы в каталог обмена с программой `compile`. При получении результата компиляции от программы `compile` статус может быть изменён на `COMPILED`, `COMPILE_ERR` или `CHECK_FAILED`.

Если сервер компиляции `compile` не запущен, файл со служебной информацией для компиляции имеет неправильный формат, или сервер компиляции не может записать файл с результатом компиляции в каталог обмена, послылка может оставаться в состоянии `COMPILING` неограниченно долго. В этом случае необходимо вручную пересудить эту послылку.

- **AVAILABLE.** Такой статус послылки устанавливается непосредственно после приёма послылки от клиента и до отправки программы на компиляцию и тестирование, если послылка проверяется автоматически (то есть не выполняются условия для установки статуса `IGNORED` или `ACCEPTED`).

После того, как файл с текстом программы будет скопирован в каталог обмена с программой `compile`, у послылки будет установлен статус `COMPILING`.

# Приложение А

## Лицензионные соглашения

### A.1 GNU Free Documentation License

Version 1.1, March 2000

Copyright © 2000 Free Software Foundation, Inc.

59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

#### Preamble

The purpose of this License is to make a manual, textbook, or other written document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

#### A.1.1 Applicability and Definitions

This License applies to any manual or other work that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall

directly within that overall subject. (For example, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, whose contents can be viewed and edited directly and straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup has been designed to thwart or discourage subsequent modification by readers is not Transparent. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, L<sup>A</sup>T<sub>E</sub>X input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML designed for human modification. Opaque formats include PostScript, PDF, proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

### **A.1.2 Verbatim Copying**

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

### **A.1.3 Copying in Quantity**

If you publish printed copies of the Document numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers,

as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a publicly-accessible computer-network location containing a complete Transparent copy of the Document, free of added material, which the general network-using public has access to download anonymously at no charge using public-standard network protocols. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

#### **A.1.4 Modifications**

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has less than five).
- State on the Title page the name of the publisher of the Modified Version, as the publisher.
- Preserve all the copyright notices of the Document.
- Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- Include an unaltered copy of this License.

- Preserve the section entitled “History”, and its title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section entitled “History” in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- In any section entitled “Acknowledgements” or “Dedications”, preserve the section’s title, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- Delete any section entitled “Endorsements”. Such a section may not be included in the Modified Version.
- Do not retitling any existing section as “Endorsements” or to conflict in title with any Invariant Section.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties – for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## **A.1.5 Combining Documents**

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections entitled “History” in the various original documents, forming one section entitled “History”; likewise combine any sections entitled “Acknowledgements”, and any sections entitled “Dedications”. You must delete all sections entitled “Endorsements.”

### **A.1.6 Collections of Documents**

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

### **A.1.7 Aggregation With Independent Works**

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, does not as a whole count as a Modified Version of the Document, provided no compilation copyright is claimed for the compilation. Such a compilation is called an “aggregate”, and this License does not apply to the other self-contained works thus compiled with the Document, on account of their being thus compiled, if they are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one quarter of the entire aggregate, the Document’s Cover Texts may be placed on covers that surround only the Document within the aggregate. Otherwise they must appear on covers around the whole aggregate.

### **A.1.8 Translation**

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License provided that you also include the original English version of this License. In case of a disagreement between the translation and the original English version of this License, the original English version will prevail.

### **A.1.9 Termination**

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is

void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

### **A.1.10 Future Revisions of This License**

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

### **ADDENDUM: How to use this License for your documents**

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have no Invariant Sections, write "with no Invariant Sections" instead of saying which ones are invariant. If you have no Front-Cover Texts, write "no Front-Cover Texts" instead of "Front-Cover Texts being LIST"; likewise for Back-Cover Texts.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

## **А.2 Перевод на русский язык Лицензии GNU на свободную документацию**

Автор перевода — Елена Тяпкина.

This is an unofficial translation of the GNU Free Documentation License (GFDL) into Russian. It was not published by the Free Software Foundation, and does not legally state the distribution terms for works that uses the GFDL — only the original English text of the GFDL does that. However, we hope that this translation will help Russian speakers understand the GFDL better.

Настоящий перевод Лицензии GNU на Свободную Документацию (GFDL) на русский язык не является официальным. Он не публикуется Free Software Foundation и не устанавливает имеющих юридическую силу условий для распространения произведений, которые распространяются на условиях GFDL. Условия, имеющие юридическую силу, закреплены исключительно в аутентичном тексте GFDL на английском языке. Я надеюсь, что настоящий перевод поможет русскоязычным пользователям лучше понять содержание GFDL.

### **GNU Free Documentation License, Версия 1.1, март 2000г.**

Copyright (C) 2000 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Каждый вправе копировать и распространять экземпляры настоящей Лицензии без внесения изменений в её текст.

### **Преамбула**

Цель настоящей Лицензии — сделать свободными справочник, руководство пользователя или иные документы в письменной форме, т.е. обеспечить каждому право свободно копировать и распространять как с изменениями, так и без изменений, за вознаграждение или бесплатно указанные документы. Настоящая Лицензия также позволяет авторам или издателям документа сохранить свою репутацию, не принимая на себя ответственность за изменения, сделанные третьими лицами.

Настоящая Лицензия относится к категории «copyleft». Это означает, что все произведения, производные от документа, должны быть свободными в соответствии с концепцией «copyleft». Настоящая Лицензия дополняет General Public License GNU, которая является лицензией «copyleft», разработанной для свободного программного обеспечения.

Настоящая Лицензия разработана для применения её к документации на свободное программное обеспечение, поскольку свободное программное обеспечение должно сопровождаться свободной документацией. Пользователь должен обладать теми же правами в отношении руководства пользователя, какими он обладает в отношении свободного программного обеспечения. При этом действие настоящей Лицензии не распространяется только на руководство пользователя. Настоящая Лицензия может применяться к любому текстовому произведению независимо от его темы или от того, издано ли данное произведение в виде печатной книги или нет. Настоящую Лицензию рекомендуется применять для произведений справочного или обучающего характера.



## А.2.1 Сфера действия, термины и их определения

Условия настоящей Лицензии применяются к любому руководству пользователя или иному произведению, которое в соответствии с уведомлением, помещённым правообладателем, может распространяться на условиях настоящей Лицензии. Далее под термином «Документ» понимается любое подобное руководство пользователя или произведение. Лицо, которому передаются права по настоящей Лицензии, в дальнейшем именуется «Лицензиат».

«Модифицированная версия Документа» — любое произведение, содержащее Документ или его часть, скопированные как с изменениями, так и без них и/или переведённые на другой язык.

«Второстепенный раздел» — имеющее название приложение или предисловие к Документу, в котором отражено исключительно отношение издателей или авторов Документа к его содержанию в целом, либо к вопросам, связанным с содержанием Документа. Второстепенный раздел не может включать в себя то, что относится непосредственно к содержанию Документа. (Например, если часть Документа является учебником по математике, во Второстепенном разделе не может содержаться что-либо имеющее отношение непосредственно к математике). Во Второстепенных разделах могут быть затронуты вопросы истории того, что составляет содержание или что связано с содержанием Документа, а также правовые, коммерческие, философские, этические или политические взгляды относительно содержания Документа.

«Неизменяемые разделы» — определённые Второстепенные разделы, названия которых перечислены как Неизменяемые разделы в уведомлении Документа, определяющем лицензионные условия.

«Текст, помещаемый на обложке» — определённые краткие строки текста, которые перечислены в уведомлении Документа, определяющем лицензионные условия, как текст, помещаемый на первой и последней страницах обложки.

«Прозрачный» экземпляр Документа — экземпляр Документа в машиночитаемой форме, представленный в формате с общедоступной спецификацией при условии, что документ может просматриваться и редактироваться непосредственно с помощью общедоступных текстовых редакторов или общедоступных программ для векторной или растровой графики (в случае, если в документе содержатся изображения векторной или растровой графики). Указанный формат должен обеспечить ввод текста Документа в программы форматирования текста или автоматический перевод Документа в различные форматы, подходящие для ввода текста Документа в программы форматирования текста. Экземпляр Документа, представленный в ином формате, разметка которого затрудняет или препятствует внесению в Документ последующих изменений пользователями, не является Прозрачным. Такой экземпляр документа называется «Непрозрачным».

Форматы, в которых может быть представлен Прозрачный экземпляр Документа, включают простой формат ASCII без разметки, формат ввода Texinfo, формат ввода LaTeX, SGML или XML с использованием общедоступного DTD, а также соответствующий стандартам простой формат HTML, предназначенный для внесения модификаций человеком. «Непрозрачные» форматы включают в себя PostScript, PDF, форматы, которые можно прочитать и редактировать только с помощью текстовых редакторов, права на использование которых свободно не передаются, форматы SGML или XML, для которых DTD или инструменты для обработки не являются общедоступными, а также генерируемый машиной HTML, который вырабатывается некоторыми текстовыми редакторами исключительно в целях вывода.

«Титульный лист» — для печатной книги собственно титульный лист, а также следующие за ним страницы, которые должны содержать сведения, помещаемые на титульном листе

в соответствии с условиями настоящей Лицензии. Для произведений, формат которых не предполагает наличие титульного листа, под Титульным листом понимается текст, который помещён перед началом основного текста произведения, после его названия, напечатанного наиболее заметным шрифтом.

### **А.2.2 Копирование без внесения изменений**

Лицензиат вправе воспроизводить и распространять экземпляры Документа на любом носителе за вознаграждение или безвозмездно при условии, что каждый экземпляр содержит текст настоящей Лицензии, знаки охраны авторских прав, а также уведомление, что экземпляр распространяется в соответствии с настоящей Лицензией, при этом Лицензиат не вправе предусматривать иные лицензионные условия дополнительно к тем, которые закреплены в настоящей Лицензии. Лицензиат не вправе использовать технические средства для воспрепятствования или контроля за чтением или последующим изготовлением копий с экземпляров, распространяемых Лицензиатом. Лицензиат вправе получать вознаграждение за изготовление и распространение экземпляров Документа. При распространении большого количества экземпляров Документа Лицензиат обязан соблюдать условия пункта 3 настоящей Лицензии.

Лицензиат вправе сдавать экземпляры Документа в прокат на условиях, определённых в предыдущем абзаце, или осуществлять публичный показ экземпляров Документа.

### **А.2.3 Тиражирование**

Если Лицензиат издаёт печатные экземпляры Документа в количестве свыше 100, и в соответствии с уведомлением Документа, определяющим лицензионные условия, Документ должен содержать Текст, помещаемый на обложке, Лицензиат обязан издавать экземпляры Документа в обложке с напечатанными на ней ясно и разборчиво соответствующими Текстами, помещаемыми на обложке: Тексты, помещаемые на первой странице обложки — на первой странице, Тексты, помещаемые на последней странице — соответственно на последней. Также на первой и последней странице обложки экземпляра Документа должно быть ясно и разборчиво указано, что Лицензиат является издателем данных экземпляров. На первой странице обложки должно быть указано полное название Документа без пропусков и сокращений, все слова в названии должны быть набраны шрифтом одинакового размера. Лицензиат вправе поместить прочие сведения на обложке экземпляра. Если при издании экземпляров Документа изменяются только сведения, помещённые на обложке экземпляра, за исключением названия Документа, и при этом соблюдаются требования настоящего пункта, такие действия приравниваются к копированию без внесения изменений.

Если объем текста, который должен быть помещён на обложке экземпляра, не позволяет напечатать его разборчиво, Лицензиат обязан поместить разумную часть текста непосредственно на обложке, а остальной текст на страницах Документа, следующих сразу за обложкой.

Если Лицензиат издаёт или распространяет Непрозрачные экземпляры Документа в количестве свыше 100, Лицензиат обязан к каждому такому экземпляру приложить Прозрачный экземпляр этого Документа в машиночитаемой форме или указать на каждом Непрозрачном экземпляре Документа адрес в компьютерной сети общего пользования, где содержится Прозрачный экземпляр без каких-либо добавленных материалов, полный текст которого каждый пользователь компьютерной сети общего пользования вправе бесплатно, не называя своего

имени и не регистрируясь, записать в память компьютера с использованием общедоступных сетевых протоколов. Во втором случае Лицензиат обязан предпринять разумные шаги с тем, чтобы доступ к Прозрачному экземпляру Документа по указанному адресу сохранялся по крайней мере в течение одного года после последнего распространения Непрозрачного экземпляра Документа данного тиража, независимо от того, было ли распространение осуществлено Лицензиатом непосредственно или через агентов или розничных продавцов.

Прежде чем начать распространение большого количества экземпляров Документа Лицензиату заблаговременно следует связаться с авторами Документа, чтобы они имели возможность предоставить Лицензиату обновлённую версию Документа. Лицензиат не обязан выполнять данное условие.

#### **А.2.4 Внесение изменений**

Лицензиат вправе воспроизводить и распространять Модифицированные версии Документа в соответствии с условиями пунктов 2 и 3 настоящей Лицензии, при условии что Модифицированная версия Документа публикуется в соответствии с настоящей Лицензией. В частности, Лицензиат обязан передать каждому обладателю экземпляра Модифицированной версии Документа права на распространение и внесение изменений в данную Модифицированную версию Документа, аналогично правам на распространение и внесение изменений, которые передаются обладателю экземпляра Документа. При распространении Модифицированных версий Документа Лицензиат обязан:

- А. поместить на Титульном листе и на обложке при её наличии название Модифицированной версии, отличающееся от названия Документа и названий предыдущих версий. Названия предыдущих версий при их наличии должны быть указаны в Документе в разделе «История». Лицензиат вправе использовать название предыдущей версии Документа с согласия издателя предыдущей версии;
- В. указать на Титульном листе в качестве авторов тех лиц, которые являются авторами изменений в Модифицированной версии, а также не менее пяти основных авторов Документа либо всех авторов, если их не более пяти;
- С. указать на Титульном листе наименование издателя Модифицированной версии, с указанием, что он является издателем данной Версии;
- Д. сохранить все знаки охраны авторского права Документа;
- Е. поместить соответствующий знак охраны авторского права на внесённые Лицензиатом изменения рядом с прочими знаками охраны авторского права;
- Ф. поместить непосредственно после знаков охраны авторского права уведомление, в соответствии с которым каждому предоставляется право использовать Модифицированную Версию в соответствии с условиями настоящей Лицензии. Текст уведомления приводится в Приложении к настоящей Лицензии;
- Г. сохранить в уведомлении, указанном в подпункте Г, полный список Неизменяемых разделов и Текста, помещаемого на обложке, перечисленных в уведомлении Документа;
- Н. включить в Модифицированную версию текст настоящей Лицензии без каких-либо изменений;

- I. сохранить в Модифицированной версии раздел «История», включая его название, и дополнить его пунктом, в котором указать так же, как данные сведения указаны на Титульном листе, название, год публикации, наименования новых авторов и издателя Модифицированной версии. Если в Документе отсутствует раздел «История», Лицензиат обязан создать в Модифицированной версии такой раздел, указать в нем название, год публикации, авторов и издателя Документа так же, как данные сведения указаны на Титульном листе Документа и дополнить этот раздел пунктом, содержание которого описано в предыдущем предложении;
- J. сохранить в Модифицированной версии адрес в компьютерной сети, указанный в Документе, по которому каждый вправе осуществить доступ к Прозрачному экземпляру Документа, а также адрес в компьютерной сети, указанный в Документе, по которому можно получить доступ к предыдущим версиям Документа. Адреса, по которым находятся предыдущие версии Документа, можно поместить в раздел «История». Лицензиат вправе не указывать адрес произведения в компьютерной сети, которое было опубликовано не менее чем за четыре года до публикации самого Документа. Лицензиат вправе не указывать адрес определённой версии в компьютерной сети с разрешения первоначального издателя данной версии;
- K. сохранить без изменений названия разделов «Благодарности» или «Посвящения», а также содержание и стиль каждой благодарности и/или посвящения;
- L. сохранить без изменений названия и содержание всех Неизменяемых разделов Документа. Нумерация данных разделов или иной способ их перечисления не включается в состав названий разделов;
- M. удалить существующий раздел Документа под названием «Одобрения». Такой раздел не может быть включён в Модифицированную версию;
- N. не присваивать существующим разделам Модифицированной версии название «Одобрения» или такие названия, которые повторяют название любого из Неизменяемых разделов.

Если в Модифицированную версию включены новые предисловия или приложения, которые могут быть определены как Второстепенные разделы и которые не содержат текст, скопированный из Документа, Лицензиат вправе по своему выбору определить все или некоторые из этих разделов как Неизменяемые. Для этого следует добавить их названия в список Неизменяемых разделов в уведомлении в Модифицированной версии, определяющем лицензионные условия. Названия данных разделов должны отличаться от названий всех остальных разделов.

Лицензиат вправе дополнить Модифицированную версию новым разделом «Одобрения» при условии, что в него включены исключительно одобрения Модифицированной версии Лицензиата третьими сторонами, например оценки экспертов или указания, что текст Модифицированной версии был одобрен организацией в качестве официального определения стандарта.

Лицензиат вправе дополнительно поместить на обложке Модифицированной версии Текст, помещаемый на обложке, не превышающий пяти слов для первой страницы обложки и 25 слов для последней страницы обложки. К Тексту, помещаемому на обложке, каждым лицом непосредственно или от имени этого лица на основании соглашения с ним может быть добавлено только по одной строке на первой и на последней страницах обложки. Если

на обложке Документа Лицензиатом от своего имени или от имени лица, в интересах которого действует Лицензиат, уже был помещён Текст, помещаемый на обложке, Лицензиат не вправе добавить другой Текст. В этом случае Лицензиат вправе заменить старый текст на новый с разрешения предыдущего издателя, который включил старый текст в издание.

По настоящей Лицензии автор(ы) и издатель(и) Документа не передают право использовать их имена и/или наименования в целях рекламы или заявления или предположения, что любая из Модифицированных Версий получила их одобрение.

### **А.2.5 Объединение документов**

Лицензиат с соблюдением условий п.4 настоящей Лицензии вправе объединить Документ с другими документами, которые опубликованы на условиях настоящей Лицензии, при этом Лицензиат должен включить в произведение, возникшее в результате объединения, все Неизменяемые разделы из всех первоначальных документов без внесения в них изменений, а также указать их в качестве Неизменяемых разделов данного произведения в списке Неизменяемых разделов, который содержится в уведомлении, определяющем лицензионные условия для произведения.

Произведение, возникшее в результате объединения, должно содержать только один экземпляр настоящей Лицензии. Повторяющиеся в произведении одинаковые Неизменяемые разделы могут быть заменены единственной копией таких разделов. Если произведение содержит несколько Неизменяемых Разделов с одним и тем же названием, но с разным содержанием, Лицензиат обязан сделать название каждого такого раздела уникальным путём добавления после названия в скобках уникального номера данного раздела или имени первоначального автора или издателя данного раздела, если автор или издатель известны Лицензиату. Лицензиат обязан соответственно изменить названия Неизменяемых разделов в списке Неизменяемых разделов в уведомлении, определяющем лицензионные условия для произведения, возникшего в результате объединения.

В произведении, возникшем в результате объединения, Лицензиат обязан объединить все разделы «История» из различных первоначальных Документов в один общий раздел «История». Подобным образом Лицензиат обязан объединить все разделы с названием «Благодарности» и «Посвящения». Лицензиат обязан исключить из произведения все разделы под названием «Одобрения».

### **А.2.6 Сборники документов**

Лицензиат вправе издать сборник, состоящий из Документа и других документов, публикуемых в соответствии с условиями настоящей Лицензии. В этом случае Лицензиат вправе заменить все экземпляры настоящей Лицензии в документах одним экземпляром, включённым в сборник, при условии, что остальной текст каждого документа включён в сборник с соблюдением условий по осуществлению копирования без внесения изменений.

Лицензиат вправе выделить какой-либо документ из сборника и издать его отдельно в соответствии с настоящей Лицензией, при условии, что Лицензиатом в данный документ включён текст настоящей Лицензии и им соблюдены условия Лицензии по осуществлению копирования без внесения изменений в отношении данного документа.

## **А.2.7 Подборка документа и самостоятельных произведений**

Размещение Документа или произведений, производных от Документа, с другими самостоятельными документами или произведениями на одном устройстве для хранения информации или носителе не влечёт за собой возникновения Модифицированной версии Документа, при условии, что Лицензиат не заявляет авторских прав на осуществлённый им подбор или расположение документов при их размещении. Такое размещение называется «Подборкой», при этом условия настоящей Лицензии не применяются к самостоятельным произведениям, размещённым вышеуказанным способом вместе с Документом, при условии, что они не являются произведениями, производными от Документа.

Если условия пункта 3 настоящей Лицензии относительно Текста, помещаемого на обложке, могут быть применены к экземплярам Документа в Подборке, то в этом случае Текст с обложки Документа может быть помещён на обложке только собственно Документа внутри подборки при условии, что Документ занимает менее четвертой части объёма всей Подборки. Если Документ занимает более четвертой части объёма Подборки, в этом случае Текст с обложки Документа должен быть помещён на обложке всей Подборки.

## **А.2.8 Перевод**

Перевод является одним из способов модификации Документа, в силу чего Лицензиат вправе распространять экземпляры перевода Документа в соответствии с пунктом 4 настоящей Лицензии. Замена Неизменяемых разделов их переводами может быть осуществлена только с разрешения соответствующих правообладателей, однако Лицензиат вправе в дополнение к оригинальным версиям таких Неизменяемых разделов включить в текст экземпляра перевод всех или части таких Разделов. Лицензиат вправе включить в текст экземпляра перевод настоящей Лицензии при условии, что в него включён также и оригинальный текст настоящей Лицензии на английском языке. В случае разногласий в толковании текста перевода и текста на английском языке предпочтение отдаётся тексту Лицензии на английском языке.

## **А.2.9 Расторжение лицензии**

Лицензиат вправе воспроизводить, модифицировать, распространять или передавать права на использование Документа только на условиях настоящей Лицензии. Любое воспроизведение, модификация, распространение или передача прав на иных условиях являются недействительными и автоматически ведут к расторжению настоящей Лицензии и прекращению всех прав Лицензиата, предоставленных ему настоящей Лицензией. При этом права третьих лиц, которым Лицензиат в соответствии с настоящей Лицензией передал экземпляры Документа или права на него, сохраняются в силе при условии полного соблюдения ими настоящей Лицензии.

## **А.2.10 Пересмотр условий лицензии**

Free Software Foundation может публиковать новые исправленные версии GFDL. Такие версии могут быть дополнены различными нормами, регулирующими правоотношения, которые возникли после опубликования предыдущих версий, однако в них будут сохранены основные принципы, закреплённые в настоящей версии.

Каждой версии присваивается свой собственный номер. Если указано, что Документ распространяется в соответствии с определённой версией, т.е. указан её номер, или любой более поздней версией настоящей Лицензии, Лицензиат вправе присоединиться к любой из этих версий Лицензии, опубликованных Free Software Foundation (при условии, что ни одна из версий не является проектом Лицензии). Если Документ не содержит такого указания на номер версии Лицензии Лицензиат вправе присоединиться к любой из версий Лицензии, опубликованных когда-либо Free Software Foundation (при условии, что ни одна из версий не является Проектом Лицензии).

## **А.2.11 Порядок применения условий настоящей Лицензии**

Чтобы применить условия настоящей Лицензии к созданному вами документу, вам следует включить в документ текст настоящей Лицензии, а также знак охраны авторского права и уведомление, определяющее лицензионные условия, сразу после титульного листа документа в соответствии с нижеприведённым образцом:

© имя (наименование) автора или иного правообладателя, год первого опубликования документа. Каждый имеет право воспроизводить, распространять и/или вносить изменения в настоящий Документ в соответствии с условиями GNU Free Documentation License, Версией 1.1 или любой более поздней версией, опубликованной Free Software Foundation. Данный Документ содержит следующие Неизменяемые разделы (указать названия Неизменяемых разделов). данный документ содержит следующий Текст, помещаемый на первой странице обложки (перечислить), данный документ содержит следующий Текст, помещаемый на последней странице обложки (перечислить). Копия настоящей Лицензии включена в раздел под названием «GNU Free Documentation License».

Если документ не содержит Неизменяемых разделов, укажите «Данный документ не содержит Неизменяемых разделов». Если документ не содержит Текста, помещаемого на первой или последней страницах обложки, укажите «Данный документ не содержит Текста, помещаемого на первой странице обложки», соответственно укажите для последней страницы обложки.

Если ваш документ содержит имеющие существенное значение примеры программного кода, мы рекомендуем вам выпустить их отдельно в соответствии с условиями одной из лицензий на свободное программное обеспечение, например GNU General Public License, чтобы их можно было использовать как свободное программное обеспечение.

## A.3 GNU GENERAL PUBLIC LICENSE

Version 2, June 1991.

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software — to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation’s software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author’s protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors’ reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone’s free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

### A.3.1 Terms and conditions for copying, distribution and modification

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The “Program”, below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.



Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

- (a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.
- (b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.
- (c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:
  - Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
  - Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,
  - Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.
6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

## **NO WARRANTY**

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE

LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS"WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

### **A.3.2 How to Apply These Terms to Your New Programs**

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright"line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C)  
19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type 'show w'. This is free software,

and you are welcome to redistribute it under certain conditions; type 'show c' for details.

The hypothetical commands 'show w' and 'show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than 'show w' and 'show c'; they could even be mouse-clicks or menu items — whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program 'Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

## A.4 Перевод на русский язык GNU General Public License

Автор перевода — Елена Тяпкина.

This is an unofficial translation of the GNU General Public License into Russian. It was not published by the Free Software Foundation, and does not legally state the distribution terms for software that uses the GNU GPL — only the original English text of the GNU GPL does that. However, we hope that this translation will help Russian speakers understand the GNU GPL better.

Настоящий перевод Стандартной Общественной Лицензии GNU на русский язык не является официальным. Он не публикуется Free Software Foundation и не устанавливает имеющих юридическую силу условий для распространения программного обеспечения, которое распространяется на условиях Стандартной Общественной Лицензии GNU. Условия, имеющие юридическую силу, закреплены исключительно в аутентичном тексте Стандартной Общественной Лицензии GNU на английском языке. Я надеюсь, что настоящий перевод поможет русскоязычным пользователям лучше понять содержание Стандартной Общественной Лицензии GNU.

### GNU General Public License

Версия 2, июнь 1991г.

Copyright (C) 1989, 1991 Free Software Foundation, Inc.

59 Temple Place - Suite 330, Boston, MA 02111-1307, USA

Каждый вправе копировать и распространять экземпляры настоящей Лицензии без внесения изменений в её текст.

### Преамбула

Большинство лицензий на программное обеспечение лишает вас права распространять и вносить изменения в это программное обеспечение. Стандартная Общественная Лицензия GNU, напротив, разработана с целью гарантировать вам право совместно использовать и вносить изменения в свободное программное обеспечение, т.е. обеспечить свободный доступ к программному обеспечению для всех пользователей. Условия настоящей Стандартной Общественной Лицензии применяются к большей части программного обеспечения Free Software Foundation, а также к любому другому программному обеспечению по желанию его автора. (К некоторому программному обеспечению Free Software Foundation применяются условия Стандартной Общественной Лицензии GNU для Библиотек). Вы также можете применять Стандартную Общественную Лицензию к разработанному вами программному обеспечению.

Говоря о свободном программном обеспечении, мы имеем в виду свободу, а не безвозмездность. Настоящая Стандартная Общественная Лицензия разработана с целью гарантировать вам право распространять экземпляры свободного программного обеспечения (и при желании получать за это вознаграждение), право получать исходный текст программного обеспечения или иметь возможность его получить, право вносить изменения в программное обеспечение или использовать его части в новом свободном программном обеспечении, а также право знать, что вы имеете все вышеперечисленные права.

Чтобы защитить ваши права, мы вводим ряд ограничений с тем, чтобы никто не имел возможности лишить вас этих прав или обратиться к вам с предложением отказаться от этих прав. Данные ограничения налагают на вас определённые обязанности в случае, если вы

распространяете экземпляры программного обеспечения или модифицируете программное обеспечение.

Например, если вы распространяете экземпляры такого программного обеспечения за плату или бесплатно, вы обязаны передать новым обладателям все права в том же объёме, в каком они принадлежат вам. Вы обязаны обеспечить получение новыми обладателями программы её исходного текста или возможность его получить. Вы также обязаны ознакомить их с условиями настоящей Лицензии.

Для защиты ваших прав мы:

- (1) оставляем за собой авторские права на программное обеспечение и
- (2) предлагаем вам использовать настоящую Лицензию, в соответствии с условиями которой вы вправе воспроизводить, распространять и/или модифицировать программное обеспечение.

Кроме того, для защиты как нашей репутации, так и репутации других авторов программного обеспечения, мы уведомляем всех пользователей, что на данное программное обеспечение никаких гарантий не предоставляется. Те, кто приобрёл программное обеспечение, с внесёнными в него третьими лицами изменениями, должны знать, что они получают не оригинал, в силу чего автор оригинала не несёт ответственности за ошибки в работе программного обеспечения, допущенные третьими лицами при внесении изменений.

Наконец, программное обеспечение перестаёт быть свободным в случае, если лицо приобретает на него исключительные права [1]. Недопустимо, чтобы лица, распространяющие свободное программное обеспечение, могли приобрести исключительные права на использование данного программного обеспечения и зарегистрировать их в Патентном ведомстве. Чтобы избежать этого, мы заявляем, что обладатель исключительных прав обязан предоставить любому лицу права на использование программного обеспечения либо не приобретать исключительных прав вообще.

Ниже изложены условия воспроизведения, распространения и модификации программного обеспечения.

#### **А.4.1 Условия воспроизведения, распространения и модификации**

0. Условия настоящей Лицензии применяются ко всем видам программного обеспечения или любому иному произведению, которое содержит указание правообладателя на то, что данное произведение может распространяться на условиях Стандартной Общественной Лицензии. Под термином «Программа» далее понимается любое подобное программное обеспечение или иное произведение. Под термином «произведение, производное от Программы» понимается Программа или любое иное производное произведение в соответствии с законодательством об авторском праве [2], т.е. произведение, включающее в себя Программу или её часть, как с внесёнными в её текст изменениями, так и без них и/или переведённую на другой язык. (Здесь и далее, понятие «модификация» включает в себя понятие перевода в самом широком смысле). Каждый приобретатель экземпляра Программы именуется в дальнейшем «Лицензиат».

Действие настоящей Лицензии не распространяется на осуществление иных прав, кроме воспроизведения, распространения и модификации программного обеспечения. Не устанавливается ограничений на запуск Программы. Условия Лицензии распространяются на выходные данные из Программы только в том случае, если их содержание

составляет произведение, производное от Программы (независимо от того, было ли такое произведение создано в результате запуска Программы). Это зависит от того, какие функции выполняет Программа.

1. Лицензиат вправе изготавливать и распространять экземпляры исходного текста Программы в том виде, в каком он его получил, без внесения в него изменений на любом носителе, при соблюдении следующих условий: на каждом экземпляре помещён знак охраны авторского права и уведомление об отсутствии гарантий; оставлены без изменений все уведомления, относящиеся к настоящей Лицензии и отсутствию гарантий; вместе с экземпляром Программы приобретателю передаётся копия настоящей Лицензии.

Лицензиат вправе взимать плату за передачу экземпляра Программы, а также вправе за плату оказывать услуги по гарантийной поддержке Программы.

2. Лицензиат вправе модифицировать свой экземпляр или экземпляры Программы полностью или любую её часть. Данные действия Лицензиата влекут за собой создание произведения, производного от Программы. Лицензиат вправе изготавливать и распространять экземпляры такого произведения, производного от Программы, или собственно экземпляры изменений в соответствии с пунктом 1 настоящей Лицензии при соблюдении следующих условий:

- (a) файлы, изменённые Лицензиатом, должны содержать хорошо заметную пометку, что они были изменены, а также дату внесения изменений;
- (b) при распространении или публикации Лицензиатом любого произведения, которое содержит Программу или её часть или является производным от Программы или от её части, Лицензиат обязан передавать права на использование данного произведения третьим лицам на условиях настоящей Лицензии, при этом Лицензиат не вправе требовать уплаты каких-либо лицензионных платежей. Распространяемое произведение лицензируется как одно целое;
- (c) если модифицированная Программа при запуске обычно читает команды в интерактивном режиме, Лицензиат обязан обеспечить вывод на экран дисплея или печатающее устройство сообщения, которое должно включать в себя:
  - знак охраны авторского права;
  - уведомление об отсутствии гарантий на Программу (или иное, если Лицензиат предоставляет гарантии);
  - указание на то, что пользователи вправе распространять экземпляры Программы в соответствии с условиями настоящей Лицензии, а также на то, каким образом пользователь может ознакомиться с текстом настоящей Лицензии. (Исключение: если оригинальная Программа является интерактивной, но не выводит в своём обычном режиме работы сообщение такого рода, то вывод подобного сообщения произведением, производным от Программы, в этом случае не обязателен).

Вышеуказанные условия применяются к модифицированному произведению, производному от Программы, в целом. В случае если отдельные части данного произведения не являются производными от Программы, являются результатом творческой деятельности и могут быть использованы как самостоятельное произведение, Лицензиат вправе распространять отдельно такое произведение на иных лицензионных условиях. В



случае если Лицензиат распространяет вышеуказанные части в составе произведения, производного от Программы, то условия настоящей Лицензии применяются к произведению в целом, при этом права, приобретаемые сублицензиатами на основании Лицензии, передаются им в отношении всего произведения, включая все его части, независимо от того, кто является их авторами.

Целью настоящего пункта 2 не является заявление прав или оспаривание прав на произведение, созданное исключительно Лицензиатом. Целью настоящего пункта является обеспечение права контролировать распространение произведений, производных от Программы, и составных произведений, производных от Программы.

Размещение произведения, которое не является производным от Программы, на одном устройстве для хранения информации или носителе вместе с Программой или произведением, производным от Программы, не влечёт за собой распространения условий настоящей Лицензии на такое произведение.

3. Лицензиат вправе воспроизводить и распространять экземпляры Программы или произведения, которое является производным от Программы, в соответствии с пунктом 2 настоящей Лицензии, в виде объектного кода или в исполняемой форме в соответствии с условиями п.п. 1 и 2 настоящей Лицензии при соблюдении одного из перечисленных ниже условий:
  - (а) к экземпляру должен прилагаться соответствующий полный исходный текст в машиночитаемой форме, который должен распространяться в соответствии с условиями п.п. 1 и 2 настоящей Лицензии на носителе, обычно используемом для передачи программного обеспечения, либо
  - (b) к экземпляру должно прилагаться действительное в течение трёх лет предложение в письменной форме к любому третьему лицу передать за плату, не превышающую стоимость осуществления собственно передачи, экземпляр соответствующего полного исходного текста в машиночитаемой форме в соответствии с условиями п.п. 1 и 2 настоящей Лицензии на носителе, обычно используемом для передачи программного обеспечения, либо
  - (с) к экземпляру должна прилагаться полученная Лицензиатом информация о предложении, в соответствии с которым можно получить соответствующий исходный текст. (Данное положение применяется исключительно в том случае, если Лицензиат осуществляет некоммерческое распространение программы, при этом программа была получена самим Лицензиатом в виде объектного кода или в исполняемой форме и сопровождалась предложением, соответствующим условиям пп. b п.3 настоящей Лицензии).

Под исходным текстом произведения понимается такая форма произведения, которая наиболее удобна для внесения изменений. Под полным исходным текстом исполняемого произведения понимается исходный текст всех составляющих произведение модулей, а также всех файлов, связанных с описанием интерфейса, и сценариев, предназначенных для управления компиляцией и установкой исполняемого произведения. Однако, в качестве особого исключения, распространяемый исходный текст может не включать того, что обычно распространяется (в виде исходного текста или в бинарной форме) с основными компонентами (компилятор, ядро и т.д.) операционной системы,

в которой работает исполняемое произведение, за исключением случаев, когда исполняемое произведение сопровождается таким компонентом.

В случае если произведение в виде объектного кода или в исполняемой форме распространяется путём предоставления доступа для копирования его из определённого места, обеспечение равноценного доступа для копирования исходного текста из этого же места удовлетворяет требованиям распространения исходного текста, даже если третьи лица при этом не обязаны копировать исходный текст вместе с объектным кодом произведения.

4. Лицензиат вправе воспроизводить, модифицировать, распространять или передавать права на использование Программы только на условиях настоящей Лицензии. Любое воспроизведение, модификация, распространение или передача прав на иных условиях являются недействительными и автоматически ведут к расторжению настоящей Лицензии и прекращению всех прав Лицензиата, предоставленных ему настоящей Лицензией. При этом права третьих лиц, которым Лицензиат в соответствии с настоящей Лицензией передал экземпляры Программы или права на неё, сохраняются в силе при условии полного соблюдения ими настоящей Лицензии.
5. Лицензиат не обязан присоединяться к настоящей Лицензии, поскольку он её не подписал. Однако только настоящая Лицензия предоставляет право распространять или модифицировать Программу или произведение, производное от Программы. Подобные действия нарушают действующее законодательство, если они не осуществляются в соответствии с настоящей Лицензией. Если Лицензиат внёс изменения или осуществил распространение экземпляров Программы или произведения, производного от Программы, Лицензиат тем самым подтвердил своё присоединение к настоящей Лицензии в целом, включая условия, определяющие порядок воспроизведения, распространения или модификации Программы или произведения, производного от Программы.
6. При распространении экземпляров Программы или произведения, производного от Программы, первоначальный лицензиат автоматически передаёт приобретателю такого экземпляра право воспроизводить, распространять и модифицировать Программу в соответствии с условиями настоящей Лицензии. Лицензиат не вправе ограничивать каким-либо способом осуществление приобретателями полученных ими прав. Лицензиат не несёт ответственности за несоблюдение условий настоящей Лицензии третьими лицами.
7. Лицензиат не освобождается от исполнения обязательств в соответствии с настоящей Лицензией в случае, если в результате решения суда или заявления о нарушении исключительных прав или в связи с наступлением иных обстоятельств, не связанных непосредственно с нарушением исключительных прав, на Лицензиата на основании решения суда, договора или ином основании возложены обязательства, которые противоречат условиям настоящей Лицензии. В этом случае Лицензиат не вправе распространять экземпляры Программы, если он не может одновременно исполнить условия настоящей Лицензии и возложенные на него указанным выше способом обязательства. Например, если по условиям лицензионного соглашения сублицензиатам не может быть предоставлено право бесплатного распространения экземпляров Программы, которые они приобрели напрямую или через третьих лиц у Лицензиата, то в этом случае Лицензиат обязан отказаться от распространения экземпляров Программы.

Если любое положение настоящего пункта при наступлении конкретных обстоятельств будет признано недействительным или неприменимым, настоящий пункт применяется за исключением такого положения. Настоящий пункт применяется в целом при прекращении вышеуказанных обстоятельств или их отсутствии.

Целью данного пункта не является принуждение Лицензиата к нарушению патента или заявления на иные права собственности или к оспариванию действительности такого заявления. Единственной целью данного пункта является защита неприкосновенности системы распространения свободного программного обеспечения, которая обеспечивается за счёт общественного лицензирования. Многие люди внесли свой щедрый вклад в создание большого количества программного обеспечения, которое распространяется через данную систему в надежде на её длительное и последовательное применение. Лицензиат не вправе вынуждать автора распространять программное обеспечение через данную систему. Право выбора системы распространения программного обеспечения принадлежит исключительно его автору.

Настоящий пункт 7 имеет целью чётко определить те цели, которые преследуют все остальные положения настоящей Лицензии.

8. В том случае если распространение и/или использование Программы в отдельных государствах ограничено соглашениями в области патентных или авторских прав, первоначальный правообладатель, распространяющий Программу на условиях настоящей Лицензии, вправе ограничить территорию распространения Программы, указав только те государства, на территории которых допускается распространение Программы без ограничений, обусловленных такими соглашениями. В этом случае такое указание в отношении территорий определённых государств признается одним из условий настоящей Лицензии.
9. Free Software Foundation может публиковать исправленные и/или новые версии настоящей Стандартной Общественной Лицензии. Такие версии могут быть дополнены различными нормами, регулирующими правоотношения, которые возникли после опубликования предыдущих версий, однако в них будут сохранены основные принципы, закреплённые в настоящей версии.

Каждой версии присваивается свой собственный номер. Если указано, что Программа распространяется в соответствии с определённой версией, т.е. указан её номер, или любой более поздней версией настоящей Лицензии, Лицензиат вправе присоединиться к любой из этих версий Лицензии, опубликованных Free Software Foundation. Если Программа не содержит такого указания на номер версии Лицензии Лицензиат вправе присоединиться к любой из версий Лицензии, опубликованных когда-либо Free Software Foundation.

10. В случае если Лицензиат намерен включить часть Программы в другое свободное программное обеспечение, которое распространяется на иных условиях, чем в настоящей Лицензии, ему следует испросить письменное разрешение на это у автора программного обеспечения. Разрешение в отношении программного обеспечения, права на которое принадлежат Free Software Foundation, следует испрашивать у Free Software Foundation. В некоторых случаях Free Software Foundation делает исключения. При принятии решения Free Software Foundation будет руководствоваться двумя целями: сохранение статуса свободного для любого произведения, производного от свободного программного

обеспечения Free Software Foundation и обеспечение наиболее широкого совместного использования программного обеспечения.

## **ОТСУТСТВИЕ ГАРАНТИЙНЫХ ОБЯЗАТЕЛЬСТВ**

11. ПОСКОЛЬКУ НАСТОЯЩАЯ ПРОГРАММА РАСПРОСТРАНЯЕТСЯ БЕСПЛАТНО, ГАРАНТИИ НА НЕЁ НЕ ПРЕДОСТАВЛЯЮТСЯ В ТОЙ СТЕПЕНИ, В КАКОЙ ЭТО ДОПУСКАЕТСЯ ПРИМЕНИМЫМ ПРАВОМ. НАСТОЯЩАЯ ПРОГРАММА ПОСТАВЛЯЕТСЯ НА УСЛОВИЯХ «КАК ЕСТЬ». ЕСЛИ ИНОЕ НЕ УКАЗАНО В ПИСЬМЕННОЙ ФОРМЕ, АВТОР И/ИЛИ ИНОЙ ПРАВООБЛАДАТЕЛЬ НЕ ПРИНИМАЕТ НА СЕБЯ НИКАКИХ ГАРАНТИЙНЫХ ОБЯЗАТЕЛЬСТВ, КАК ЯВНО ВЫРАЖЕННЫХ, ТАК И ПОДРАЗУМЕВАЕМЫХ, В ОТНОШЕНИИ ПРОГРАММЫ, В ТОМ ЧИСЛЕ ПОДРАЗУМЕВАЕМУЮ ГАРАНТИЮ ТОВАРНОГО СОСТОЯНИЯ ПРИ ПРОДАЖЕ И ПРИГОДНОСТИ ДЛЯ ИСПОЛЬЗОВАНИЯ В КОНКРЕТНЫХ ЦЕЛЯХ, А ТАКЖЕ ЛЮБЫЕ ИНЫЕ ГАРАНТИИ. ВСЕ РИСКИ, СВЯЗАННЫЕ С КАЧЕСТВОМ И ПРОИЗВОДИТЕЛЬНОСТЬЮ ПРОГРАММЫ, НЕСЁТ ЛИЦЕНЗИАТ. В СЛУЧАЕ ЕСЛИ В ПРОГРАММЕ БУДУТ ОБНАРУЖЕНЫ НЕДОСТАТКИ, ВСЕ РАСХОДЫ, СВЯЗАННЫЕ С ТЕХНИЧЕСКИМ ОБСЛУЖИВАНИЕМ, РЕМОНТОМ ИЛИ ИСПРАВЛЕНИЕМ ПРОГРАММЫ, НЕСЁТ ЛИЦЕНЗИАТ.
12. ЕСЛИ ИНОЕ НЕ ПРЕДУСМОТРЕНО ПРИМЕНЯЕМЫМ ПРАВОМ ИЛИ НЕ СОГЛАСОВАНО СТОРОНАМИ В ДОГОВОРЕ В ПИСЬМЕННОЙ ФОРМЕ, АВТОР И/ИЛИ ИНОЙ ПРАВООБЛАДАТЕЛЬ, КОТОРЫЙ МОДИФИЦИРУЕТ И/ИЛИ РАСПРОСТРАНЯЕТ ПРОГРАММУ НА УСЛОВИЯХ НАСТОЯЩЕЙ ЛИЦЕНЗИИ, НЕ НЕСЁТ ОТВЕТСТВЕННОСТИ ПЕРЕД ЛИЦЕНЗИАТОМ ЗА УБЫТКИ, ВКЛЮЧАЯ ОБЩИЕ, РЕАЛЬНЫЕ, ПРЕДВИДИМЫЕ И КОСВЕННЫЕ УБЫТКИ (В ТОМ ЧИСЛЕ УТРАТУ ИЛИ ИСКАЖЕНИЕ ИНФОРМАЦИИ, УБЫТКИ, ПОНЕСЁННЫЕ ЛИЦЕНЗИАТОМ ИЛИ ТРЕТЬИМИ ЛИЦАМИ, НЕВОЗМОЖНОСТЬ РАБОТЫ ПРОГРАММЫ С ЛЮБОЙ ДРУГОЙ ПРОГРАММОЙ И ИНЫЕ УБЫТКИ). АВТОР И/ИЛИ ИНОЙ ПРАВООБЛАДАТЕЛЬ В СООТВЕТСТВИИ С НАСТОЯЩИМ ПУНКТОМ НЕ НЕСУТ ОТВЕТСТВЕННОСТИ ДАЖЕ В ТОМ СЛУЧАЕ, ОНИ БЫЛИ ПРЕДУПРЕЖДЕНЫ О ВОЗМОЖНОСТИ ВОЗНИКНОВЕНИЯ ТАКИХ УБЫТКОВ.

### **А.4.2 Порядок применения условий настоящей Лицензии**

Если вы создали новую программу и хотите, чтобы она принесла наибольшую пользу обществу, лучший способ достичь этого — сделать вашу программу свободной, когда каждый сможет распространять её и вносить в неё изменения в соответствии с условиями настоящей Лицензии.

В этих целях Программа должна содержать приведённое ниже уведомление. Наиболее правильным будет поместить его в начале исходного текста каждого файла для максимально ясного указания на то, что гарантии на данную программу не предоставляются. Каждый файл в любом случае должен содержать знак охраны авторского права и пояснение, где можно ознакомиться с полным текстом уведомления.

[одна строка с наименованием Программы и кратким описанием её назначения]  
(C) имя (наименование) автора или иного правообладателя, год первого опубликования программы

Данная программа является свободным программным обеспечением. Вы вправе распространять её и/или модифицировать в соответствии с условиями версии 2 либо по вашему выбору с условиями более поздней версии Стандартной Общественной Лицензии GNU, опубликованной Free Software Foundation.

Мы распространяем эту программу в надежде на то, что она будет вам полезной, однако НЕ ПРЕДОСТАВЛЯЕМ НА НЕЁ НИКАКИХ ГАРАНТИЙ, в том числе ГАРАНТИИ ТОВАРНОГО СОСТОЯНИЯ ПРИ ПРОДАЖЕ и ПРИГОДНОСТИ ДЛЯ ИСПОЛЬЗОВАНИЯ В КОНКРЕТНЫХ ЦЕЛЯХ. Для получения более подробной информации ознакомьтесь со Стандартной Общественной Лицензией GNU.

Вместе с данной программой вы должны были получить экземпляр Стандартной Общественной Лицензии GNU. Если вы его не получили, сообщите об этом в Free Software Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307, USA.

Также укажите, как можно связаться с вами по электронной или обычной почте.

Если программа работает в интерактивном режиме, сделайте так, чтобы при запуске в интерактивном режиме выводилось короткое сообщение в соответствии с образцом:

Gnomovision version 69, (C) имя автора, год первого опубликования программы  
Gnomovision распространяется БЕЗ ВСЯКИХ ГАРАНТИЙ; чтобы ознакомиться с более подробной информацией, наберите "show w". Данная программа является свободным программным обеспечением и вы можете распространять её в соответствии с условиями Стандартной Общественной Лицензии GNU. Для получения более подробной информации, наберите "show c".

При введении предлагаемых команд "show w" и "show c" на экран должны выводиться соответствующие пункты Стандартной Общественной Лицензии. Не обязательно использовать именно команды "show w" и "show c". В зависимости от функций программы, команды могут вызываться нажатием кнопки мыши или быть добавлены в меню программы.

Если вы создали программу в порядке выполнения служебных обязанностей или служебного задания работодателя, вам следует получить от него в случае необходимости письменный отказ от исключительных прав на использование данной программы [3]. Нижеприведённый текст вы можете использовать в качестве образца, заменив соответствующие имена и наименования:

ЗАО «АБВ» настоящим отказывается от всех исключительных прав на использование программы для ЭВМ «Gnomovision», автором которой является Иванов Алексей Петрович, и передаёт все исключительные права на использование указанной программы её автору, Иванову Алексею Петровичу.

Подпись руководителя организации, печать, 1 января 2001г. Фамилия, Имя, Отчество], Генеральный директор

Стандартная Общественная Лицензия GNU запрещает включать вашу программу в программы, использование которых ограничено их правообладателями. Если ваша программа является библиотекой подпрограмм, вероятно, более полезным будет разрешить связывание программ, использование которых ограничено их правообладателями, с вашей библиотекой. В этом случае вам следует использовать Стандартную Общественную Лицензию GNU для Библиотек вместо настоящей Лицензии.

### А.4.3 Примечания переводчика

1. В параграфе 7 Преамбулы в английском тексте Стандартной Общественной Лицензии GNU упоминается патент на программное обеспечение (Software Patents). В начале 90х годов XX века Апелляционный суд Федерального округа США предпринял попытку установить, когда изобретение, частью которого является программное обеспечение, является патентоспособным. Суд постановил, что в этом случае следует провести экспертизу в отношении произведения в целом. Изобретение не будет признано патентоспособным, если оно представляет собой исключительно математический алгоритм. Однако, если положенный в основу изобретения способ при помощи программного обеспечения позволяет получить конкретные, промышленно применимые результаты, в этом случае изобретение является патентоспособным. В отличие от США, в РФ соответствии с Патентным законом от 23.09.1992г. не признаются патентоспособными изобретениями программы для вычислительных машин. Защита программ для ЭВМ осуществляется на основании норм законодательства об авторском праве. Исключительные права на программу для ЭВМ принадлежат автору или иному правообладателю, который приобрёл их на основании договора или ином основании, предусмотренным законом. Правообладатель всех имущественных прав на программу для ЭВМ в течение срока действия авторского права может по своему желанию зарегистрировать программу для ЭВМ путём подачи заявки в Патентное ведомство РФ.
2. Здесь имеется в виду законодательство об авторском праве США.
3. В данном абзаце в английском тексте указано, что вам следует получить письменный отказ от исключительных прав на использование созданной вами программы у вашего работодателя, если вы работаете программистом, или у учебного заведения, в котором вы обучаетесь (школа, университет, институт, колледж). В соответствии с Законом РФ «Об авторском праве и смежных правах» такой отказ следует получить только от своего работодателя. В соответствии с указанным Законом РФ авторское право на произведение, созданное в порядке выполнения служебных обязанностей или служебного задания работодателя (служебное произведение), принадлежит автору служебного произведения. Исключительные права на использование служебного произведения (в том числе программы для ЭВМ) принадлежат лицу, с которым автор состоит в трудовых отношениях (работодателю), если в договоре между ними и автором не предусмотрено иное. Данное положение не распространяется на создание в порядке выполнения служебных обязанностей или служебного задания работодателя энциклопедий, энциклопедических словарей, периодических и продолжающихся сборников научных трудов, газет, журналов и других периодических изданий. Издателю энциклопедий, энциклопедических словарей, периодических и продолжающихся изданий принадлежат исключительные права на использование таких изданий. Авторы произведений, включённых в такие издания, сохраняют исключительные права на использование своих произведений независимо от издания в целом.

My goal was not just a verbal translation of English text of GNU General Public License in Russian, but a translation, which will follow the rules of current legislation of Russian Federation on copyrights. I hope that this will help to use GNU General Public License when distributing free software in Russian Federation. Below you may find some comments (in Russian) on current legislation of Russian Federation.

Моей целью был не просто перевод Стандартной Общественной Лицензии GNU, который бы максимально точно соответствовал аутентичному тексту на английском языке, но также учитывал нормы действующего законодательства РФ об авторском праве, что увеличило бы возможность использовать Стандартную Общественную Лицензию GPL для распространения свободного программного обеспечения на территории РФ. Ниже Вы можете ознакомиться с некоторыми комментариями относительно действующего законодательства РФ.

В настоящее время на территории Российской Федерации порядок воспроизведения, распространения и модификации программного обеспечения регулируется Законом РФ «О правовой охране программ для ЭВМ и баз данных» от 23.09.1992г. №3523-1 и Законом РФ «Об авторском праве и смежных правах» от 09.07.1993г. №5351-1.

С целью наибольшего соответствия настоящего неофициального перевода Стандартной Общественной Лицензии GNU на русский язык нормам действующего законодательства РФ об авторском праве, ниже приводятся основные понятия, используемые в тексте перевода, и их определения в соответствии с указанными выше Законами РФ:

Программное обеспечение — данное понятие не применяется в указанных Законах, однако оно является наиболее общепринятым при обозначении программ для ЭВМ в переводах лицензионных соглашений, в частности Лицензионных соглашений с конечным пользователем (EULA), на русский язык. В силу этого понятие «Программное обеспечение» используется в тексте перевода для обозначения понятия «программа для ЭВМ». Под программой для ЭВМ Закон РФ понимается объективная форма представления совокупности данных и команд, предназначенных для функционирования электронных вычислительных машин (ЭВМ) и других компьютерных устройств с целью получения определённого результата, включая подготовительные материалы, полученные в ходе разработки программы для ЭВМ, и порождаемые ею аудиовизуальные отображения.

Исключительные права на использование произведения — означает право осуществлять или разрешать следующие действия: воспроизводить произведение (право на воспроизведение); распространять экземпляры произведения любым способом: продавать, сдавать в прокат и так далее (право на распространение); публично показывать произведение (право на публичный показ), переводить произведение (право на перевод); перерабатывать, аранжировать или другим образом перерабатывать произведение (право на переработку), а также иные права в соответствии с Законом РФ «Об авторском праве и смежных правах».

Исключительные (или имущественные) права на использование программы для ЭВМ — означает исключительное право осуществлять и (или) разрешать осуществление следующих действий: выпуск в свет программы для ЭВМ, воспроизведение программы для ЭВМ (полное или частичное) в любой форме, любыми способами, распространение программы для ЭВМ, модификацию программы для ЭВМ, в том числе перевод программы для ЭВМ с одного языка на другой, а также иное использование в соответствии с Законом РФ «О правовой охране программ для ЭВМ и баз данных».

Воспроизведение Программного Обеспечения — это изготовление одного или более экземпляров Программного обеспечения в любой материальной форме, а также его запись в память ЭВМ.

Модификация (переработка) Программного Обеспечения — любые его изменения, не являющиеся адаптацией.

Распространение Программного Обеспечения — это предоставление доступа для воспроизведения в любой материальной форме Программного Обеспечения, в том числе сетевыми и иными способами, а также путём продажи, проката, сдачи в наём, предоставление взаймы, включая импорт для любой из этих целей.

