Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

Факультет математики и информатики
Кафедра информационных и управляющих систем
Направление подготовки 09.04.04 – Программная инженерия
Магистерская программа Управление разработкой программного обеспечения

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ИУС
_____ А.В. Бушманов
«14» 07 2020 г.

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

на тему: Разработка тестирующей подсистемы для автоматической проверки задач ГАУ ДПО «АмИРО» муниципального этапа Всероссийской Олимпиады по Амурской области

Исполнитель студент группы 857ом	23.06.20 (подпись, дата)	Г.В. Золотарёва
Руководитель доцент,канд. техн.наук	29.06.20 (подпись, дата)	Н.П. Семичевская
Руководитель научного содержания программы магистратуры	10.07.2020 (подпись, дата)	И.Е. Еремин
Нормоконтроль	10.07.2020 (подпись, дата)	В.В. Еремина
Рецензент	10.07.20 (подпись, дата)	О.Г. Какаулин

Благовещенск 2020

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
(ФГБОУ ВО «АмГУ»)

УТВЕРЖДАЮ
Заведующий кафедрой ИУС
_____ А.В. Бушманов
«03» 02 2020 г.

ЗАДАНИЕ

К выпускной квалификационной работе студентки Золотарёвой Галины Викторовны

Тема выпускной квалификационной работы: Разработка тестирующей подсистемы для автоматической проверки задач ГАУ ДПО «АмИРО» муниципального этапа Всероссийской Олимпиады по Амурской области

(утверждена приказом от 30.04.2020 № 810-уч)

2. Срок сдачи студентом законченной работы (проекта) 20.06.2020 г.

3. Исходные данные к выпускной квалификационной работе ТЗ

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов): определить перечень типовых информационных объектов и процессов, а также связей между ними в тестирующей подсистеме Ejudge; выполнить практическую реализацию предложенной тестирующей подсистемы Ejudge поддерживающая наиболее популярные языки программирования, используемых на олимпиадах - Pascal, C, C++, Java, Python, C#

5. Перечень материалов приложения: (наличие чертежей, таблиц, графиков, схем, программных продуктов, иллюстративного материала и т.п.) справка о внедрении, свидетельство о публикации, сертификат о публикации в журнале

6. Консультанты по выпускной квалификационной работе (с указанием относящихся к ним разделов

7. Дата выдачи задания 03.02.2020 г.

Руководитель выпускной квалификационной работы: Семичевская Наталья Петровна канд.техн.наук, доцент

(фамилия, имя, отчество, должность, ученая степень, ученое звание)

Задание принял к исполнению (дата):03.02.2020 г.

РЕФЕРАТ

Магистерская диссертация содержит – 91 с., 49 рисунков, 12 таблиц, 4 приложения, 30 источников.

ИКТ, ТЕСТИРУЮЩАЯ ПОДСИСТЕМА, ОЛИМПИАДНЫЕ ЗАДАЧИ, EJUDGE

Актуальность использование информационных технологий в учебном процессе расширяет возможности средств обучения. С каждым годом все больше программных средств применения развития новых технологий и информатизации общества.

Введение цифровой системы в процесс проведения олимпиад позволит уменьшить влияние человеческого фактора на подведение итогов работ, так же уменьшит время проверки заданий. Кроме того, на текущий момент имеется проблема на разных уровнях подготовки к решению заданий в зависимости от муниципалитета, и это кардинально отличаются от заданий областного уровня, ввод же данной системы позволит сразу отсеять участников с низким уровнем, не прибегая к затратам бюджетных средств на выезд на региональный тур.

Целью диссертационного исследования является обеспечение программным средством тестирующей подсистемы, предназначенное для автоматической проверки задач по информатике.

Объектом исследования работы является контроль знаний олимпиады муниципального этапа по информатике в Амурской области.

Научная новизна работы заключается в следующем:

- предложена тестирующая подсистема для автоматической проверки задач по информатики, поддерживающая языки программирования C++, ABC, Python;
- предложен автоматический метод генерации тестов оперативно

генерировать заданное количество тестов и управлять их в автоматическом режиме;

- предложен новый подход, который позволяет проектировать непредусмотренные интерфейсы для различных пользователей.

Практическая ценность результатов работы заключается в том, что разработана тестирующая подсистема для прохождении муниципального этапа олимпиады по информатики в Амурской области. Использование которой показало эффективность предполагаемого подхода, удовлетворяющая выявленным требования для создания интерфейса.

Это позволяет существенно повысить качество разработанной подсистемы, а также сократить показатели обнаруженных в процессе тестирования ошибок. На текущий момент подсистема работает в Oracle VM VirtualBox.

Защищаемые положения:

1. Определен перечень типовых информационных объектов и процессов, а также связей между ними в тестирующей подсистеме Ejudge.

2. Выполнена практическая реализация предложенной тестирующей подсистемы Ejudge поддерживающая наиболее популярные языки программирования, используемых на олимпиадах - Pascal, C, C++, Java, Python, C#.

По теме диссертации опубликовано 3 работы в том числе: первая статья в журнале Вестник науки и образования №22 (76) ноябрь 2019, Часть 1: Москва, вторая статья Молодежь XXI века: шаг в будущее: материалы XX региональной научно - практической конференции (23 мая 2019 г., Благовещенск): в 3 томах. – Благовещенск: Издательство Амурского гос. ун-та, 2019

СОДЕРЖАНИЕ

НОРМАТИВНЫЕ ССЫЛКИ

В настоящей выпускной квалификационной работе использованы ссылки на следующие стандарты и нормативные документы.

ГОСТ 12.2.032-78 Рабочее место при выполнении работ сидя. Общие эргономические требования

ГОСТ 19.201-78 ЕСПД Техническое задание, требования к содержанию и оформлению

ГОСТ 12.1.003-83 ССБТ Общие требования безопасности

ГОСТ 34.201-89 Информационная технология. Комплекс стандартов на автоматизированные системы

ГОСТ 34.602-89 Техническое задание на создание автоматизированной системы

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ, СОКРАЩЕНИЯ

ИКТ– Информационно коммуникационные компетенции;

СУБД – Система управления базами данных;

ОС – Операционная система;

БД – База данных;

RAD – Rapid Application Development;

ЖЦ – Жизненный цикл.

ВВЕДЕНИЕ

В сфере образования в последние годы наблюдается интеллектуализация многих сфер деятельности. Оценки уровня знаний обучаемых одно из направлений автоматизация процессов.

Заключается особенность данных процессов: во-первых присущих отдельному преподавателю в уникальности систем оценки качества обучения; во-вторых в учебном процессе в целях их использования; в третьих в параметрах в самостоятельной работе обучаемых.

В Государственном автономное образовательном учреждении дополнительного профессионального образования «Амурский областной институт развития образования» для оценки уровня существует потребность сформированности информационной и коммуникационной компетентности (ИКТ – компетентности) для обучающихся по Амурской области в разработке и внедрении в образовательный процесс средств автоматизации.

В образовательный процесс внедрение средств автоматизации позволит:

– формировать оценочные электронные тестовые задания;

– информацию о результатах оценки уровня оперативно получать сформированности ИКТ – компетентности;

– результаты оценки анализировать по различным критериям.

В первую очередь, на сервер присланного участником обрабатывающую информацию тестирующая система призвана автоматизировать процесс проверки решения, что, безусловно, сохраняет динамическую составляющую олимпиады и ускоряет подведение итогов. Вследствие того, участник соревнования может сразу узнать свой результат, что больше не будет требоваться ручная проверка и сравнить его с результатами других участников. Во-вторых, возлагая всю ответственность на машину, система позволяет отстранить людей от проверки решений, что исключает фактора влияние человека (человек для личной выгоды может выставить неверный результат или ошибиться намеренно). Ввиду того, важно наличие постоянной обратной связи участвующих в

олимпиаде количество человек, может доходить до нескольких тысяч возможности задать вопрос и быстро получить ответ. Нежелательные последствия после внедрения системы может повлечь за собой. Например, которой вердикт будет неверным в системе может возникнуть ошибка и баллы будут вычислены некорректно. Иногда участники могут воспользоваться проблемой уязвимости и обмануть систему, что позволит им изменить ход олимпиады. Таким образом, тестирующая система проектируется, чтобы избежать утечек по безопасности

Данная работа посвящена разработке клиент – сервер приложения автоматизированной системы для проведения олимпиад по информатике.

Целью диссертационного исследования является обеспечение программным средством тестирующей подсистемы, предназначенное для автоматической проверки задач по информатике.

Для достижения поставленной цели необходимо решить следующие задачи исследования:

1. Анализировать существующие программное обеспечение в исследуемой области;

2. Разработать подсистему для автоматической проверки заданий;

3. Оценить качество работы разработанного программного обеспечения для проведения олимпиад по информатике с использованием автоматической системы управления по Амурской области;

4. Выбрать средства разработки информационной системы;

5. Разработать техническую документацию.

Объектом исследования данной работы является контроль знаний олимпиады муниципального этапа по информатике в Амурской области.

Предметом исследования данной работы является особенности эффективности подсистемы для автоматической проверки качества знаний.

Гипотеза исследования состоит в том, что с помощью современных средств передачи данных сократятся трудовые затраты и вместе с тем повыситься надежность, скорость и оперативность работы.

1 ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ТЕСТИРОВАНИЯ

1.1 Анализ автоматизированных систем существующих при поддержки проведения олимпиад Ejudge, Contester

На текущий момент информационные системы для учебных целей развиваются неоспоримо быстро. В связи с этим происходит реконструкция учебного процесса и внедрение новых технологий, которые помогают улучшить образовательные процессы, к примеру, использование автоматизированной системы для проведения олимпиад.

Наибольшую полезность приносит автоматизированная проверка работ. Это позволяет сжать время, увеличить объем проверяемых работ, так же это дает возможность безошибочно осуществлять проверку.

В небольшом количестве случаев автоматизированная система проверяет решений подробнее, чем преподаватель. Это обусловлено тем, что каждая программа имеет большое количество, написанных для нее тестов. Так же система сохраняет все решения, которые были в ней записаны. И составители заданий по олимпиадам могут учесть прошлые испытания и на основе их выявить слабые места в организации условия задачи. То есть данный подход дает создателям олимпиад более полно изложить условие, что приведет к большему количеству сданных решений, уменьшить количество плохо сформулированных условий задач и более четко изложить условия задач.

Необходимо учесть, что программы по автоматизированному тестированию не могут в полной мере заменить человека. Так как они не могут проверить такие факторы как, например название переменных, соответствуют ли они пониманию программного кода без условия задачи. В то время как программа может только отличить кириллицу от латиницы. Соответствие красивому стилю написания можно лишь отчасти проверить, например в языке Python соответствует ли она PEP-8 стандарту написания кода.

Таблица 1–Информационные системы поддержки проведения олимпиад

Название	Сайт	ОС	Поддержка	Лицензия	Примечания
Ejudge	http://ejudge.ru	Linux	Да	GPL	
PCMS2	http://neerc.ifmo.ru/trains/information/software.html	Windows	Да	?	Последние версии не выкладываются в свободный доступ
DOMjudge	http://domjudge.org	Linux	Нет	GPL v2	Голландия
Contester	http://contester.ru	Windows, Linux	Нет	Проприетарная	Не поддерживается с 2010 г.
PC2	http://ecs.csus.edu/pc2	Windows, Linux, Mac	Нет	Проприетарная	Используется на финале ACM ICPC
Dudge	https://github.com/DiceMaster/dudge	Windows,Linux	Да	GPL	Пока не было ни одного релиза.
Olympiads.ru	http://olympiads.ru/school/system/index.shtml	Windows	Нет	Проприетарная	Не поддерживается с 2003 года
CMS	http://cms-dev.github.io	Linux	Да	GPL-3+	
Sharif Judge	http://sharifjudge.ir	Linux	Нет	GPL v3	
Tester	http://acm.timus.ru/tester	Windows	Да	?	
ACMserver	http://acm-server.ru	Windows	Нет	?	
BOCA	http://www.ime.usp.br/~cassio/boca/	Linux	Нет	GPL v3	Используется в Latin America ACM-ICPC
Mooshak	https://mooshak.dcc.fc.up.pt	Linux	Да	Artistic license	
DMOJ	https://github.com/DMOJ/judge	?	?	GPL v3	
Acm	http://www.cs.rit.edu/~ats/projects/acm/doc/	Linux, Solaris	Нет	?	
Moe	http://www.ucw.cz/moe/	Linux			

На сегодняшний день есть много тестирующих систем, встраиваемых на различных сайтах и онлайн-сервисах или системы без использования внешней сети. Системы, можно разделить на два вида – это системы онлайн-олимпиад и системы очных олимпиад.

Основное отличие данных методов проведения олимпиад, в том, что онлайн-олимпиады проводятся круглосуточно, без участия проверяющей стороны. А очные – в определенное время с достаточным уровнем подготовки к проведению, непосредственно перед проведением. Это подготовка программного обеспечения, подготовка аудиторий и следящих за проведением олимпиады организаторов.

Приходим к выводу, что наиболее востребованными и актуальными предметами для проведения олимпиад – являются олимпиады по информатике и программированию.

Анализируя данные о сравнении информационных систем поддержки олимпиад, можно сделать такие выводы:

1. Большая часть систем не являются кросплатформенными, что сильно ограничивает их в использовании в масштабных размерах.
2. Часть систем была произведена на коммерческой основе и имеет закрытый доступ к программному коду и поддержка этих систем была прекращена.
3. Такие системы как «Ejudge», «DOMjudge», «Dudge», «Sharif Judge», «BOCA», «DMOJ» и «Moe» имеют открытый программный код и их лицензия GPL (General Public License) является общественной собственностью. Существуют программы с лицензией, которая ограничивает доступ к использованию продукта, например продукты «Contester», «PC2», «Olympiads.ru» является частной собственностью авторов или правообладателей и не отвечают требованиям свободного программного обеспечения.

Часть возникших проблем при настройке систем, обусловлены тем, что не имеют полную документацию о продукте или же документация представлена на другом языке.

В итоге, чтобы использовать тестирующую систему, преподавателю необходимо иметь навыки по работе с нею, выполнять административные задачи, задачи системного программирования, это уменьшает количество

людей, кто может проводить олимпиады со специальным программным обеспечением.

Школьники, которые изучают основы языков программирования не способны самостоятельно использовать автоматизированную систему. Для них эта задача становится невыполнимой.

На сегодняшний день из наблюдений наиболее часто использующихся систем тестирования отмечают «Ejudge» и «Contester». Эти системы закрепились в Российском образовании, как основные.

Ejudge

Данная система использует язык С. Имеет поддержку различных дистрибутивов Linux.

С помощью веб-технологий организован основной интерфейс системы, имеет расширенные возможности настроек. Работа с системой осуществляется с помощью веб-интерфейса Apache с использованием специальных настроек. С помощью утилит из командной строки и программы, которые имеют текстовый интерфейс, есть возможность реализовать дополнительные функции для администрирования.

Информацию можно получить с помощью программных средств, данные получены в виде (XML- и CSV-форматов).

Веб-интерфейс содержащий панель для администрирования содержит в себе возможность создавать и настраивать новый турнир, при этом не придется редактировать файлы для настройки программы – конфигурацию.

В системе «Ejudge» есть поддержка почти всех известных языков программирования: C, C++, Java, Pascal, Perl, Python и другие. Есть возможность добавлять языки программирования в систему тестирования.

Из всех систем, система «Ejudge» часто используется для осуществления олимпиадного тестирования по программированию. Задачи, которые стоят перед системой, успешно ею выполняются.

Недостатком системы можно назвать: трудность ее установки, сложность поддержки, некросплатформенность, ориентирована только на операционную

систему Linux, интерфейс программы не совсем логичен и удобен в использовании.

Система представляет собой продукт, который гибок и который можно расширять, его можно использовать как комплекс компонентов или брать отдельные элементы системы.

Система является достаточно сложной . Модель системы построена с упущениями.

Contester

Система «Counster» реализована на объектно-ориентированном языке Delphi и с использованием FreePascal. Работа с базами данных организована через СУБД Firebird, компоненты, которые использовались – это Indy.Socets и библиотека Zlib. Операционные системы, на которых программу можно развернуть и использовать – это Windows и Linux. Тестирующая система содержит компиляторы программ для языков C++, Object Pascal, NET, Visual Basic, C#, Java, J#.

На текущий момент базис системы «Contester» построен по правилам ACM-соревнований.

Система испытаний включает в себя HTTP-сервер, также модуль для тестирования. Система располагается в трее. Взаимодействие с системой происходит через веб-интерфейс – это прохождение заданий и отправка готовых решений.

С помощью интерфейса программы участник олимпиады может производить множество важных для него действий – это производить регистрацию в системе, работать с тестами задач, знакомиться со справочной информацией о системе, обсуждать задачи при необходимости, участвовать в форуме.

С помощью интерфейса администратора можно заниматься созданием и блокировкой, так же очищением учетных записей пользователей системы, возможно редактирование турниров, разделов и сборников. Заниматься компиляцией решений, просмотром логирования компиляции, просмотром ответов

участников олимпиады, добавлять новые языки программирования в систему, так же распределять нагрузку на сервера.

В связи, с этим не предусматривает методы для проведения и проверки заданий в форме тестов.

«Contester» содержит достаточно хороший функционал, но в нем отсутствует техническая поддержка олимпиад по программированию. Стоит отметить положительные стороны системы – это наличие установщика программы. Программу можно установить без как-либо знаний. Для нее не требуются навыки установки и настройки ПО. Что значительно отличает эту систему от «Ejudge».

Но слабой стороной системы является проприетарность.

Таблица 2 – Набор технологий для систем поддержки олимпиад по программированию «Ejudge» и «Contester»

Система	**ОС**	**СУБД**	**Среда разработки**	**Дополнительно**
Ejudge	Linux	MySQL	C, Java	Apache
Contester	Windows, Linux	Firebird	Delphi, Freepascal	Indy.Sockets, Zlib

В таблице 3 представлены положительные и отрицательные стороны систем.

Таблица 3 – Сравнение систем «Ejudge» и «Contester»

Функции	**Ejudge**	**Contester**
Несколько способов технической реализации системы	+	+
Ориентированность на различные ОС	_	+
Наличие в системе интерпретатора программного кода	+	+
Поддержка распространенных языков программирования	+	+
Автоматизация проверки вопросов в форме тестов	+	_
Простота в настройке, администрировании и эксплуатации	_	+
Поддержка ресурсов, обеспечивающих работоспособность	+	_
Обновление и поддержка системы	+	+
Свободно распространяемая лицензия	+	_
Интерфейс участника, администратора, системы	+	+
Хранение олимпиадных заданий и данных о пользователях	+	+
Сбор статистики	_	

Ни одна из систем не может полностью обеспечивать все функции в силу ограниченности возможностей данных систем, закрытости исходного кода, а также ряда других указанных выше причин, необходимые для поддержки проведения олимпиады по информатике.

Таким образом, становится понятным, что требуется спроектировать информационно-управляющую, удобную, надежную, не требующую сложной настройки систему, которая могла бы реализовать все описанные выше особенности. Информационно-управляющая система должна быть основана на применении свободных программных продуктов и использовании новейших технологий.

При разработке системы в первую очередь должен быть сделан упор на гибкость, простоту расширения, а также на развитие и интеграцию с другими системами, на максимально повторное использование уже готовых решений - на всё, что смогло бы повысить качество программного продукта и обеспечить возможность его дальнейшего развития.

1.2 Анализ предметной области

Требования, которые должна удовлетворять система автоматической проверки выполняемые функции в учебном процессе приведены в таблице 4.

Таблица 4 – Требования по программированию для проведения олимпиад к тестирующей системе

Свойство	Выполняемый функционал
Безопасность	Устойчива система к взломам; Защищены персональные данные; передается по защищенному протоколу информация
Расширяемость	Различные режимы работы поддерживает система; тестирование может быть дополнено другими видами
Скорость	В режиме работает реального времени система; сервер отправляет ответные данные и быстро обрабатывает запросы
Простота	Настраивается и легко устанавливается; клиентской части интерфейс интуитивно понятен для среднестатистического пользователя.

Текущая версия системы ejudge будет поддерживать следующие возможности:

– Основной интерфейс системы (как пользовательский, так и администраторский) реализован через WEB. Дополнительные административные возможности предоставляются через утилиты командной строки и программы с текстовым интерфейсом. Внешний вид генерируемых WEB-страниц может модифицироваться под потребности сайта. Утилиты командной строки позволяют получить информацию в формате, удобном для дальнейшей обработки (XML- и CSV-форматы).

– Многоязыковой интерфейс. Локализация выполняется с помощью соответствующего файла. Поддержка произвольных 8-битных кодировок.

В настоящее время в системе поддерживается английский и русский язык сообщений.

– Одновременная поддержка многих турниров олимпиады;

– Ведение базы пользователей с привязкой информации о пользователе к турниру олимпиады, то есть каждый турнир может требовать свой набор полей информации о пользователе, при этом значение одного и того же поля может быть разным для разных турниров;

– проведение турниров с автоматической проверкой задач по нескольким системам выставления оценки. Турниры могут быть как ограниченные, так и неограниченные по времени. Дополнительные ограничения времени могут задаваться для каждой задачи индивидуально. Время участника может быть как абсолютным, так и относительным (т.н. виртуальные турниры).

– Тонкое управление правами доступа пользователей. Некоторый пользователь может быть администратором одного турнира и не иметь никаких привилегий в другом турнире.[1][10]

– Административный WEB – интерфейс позволяет выполнить все операции по созданию и настройке нового турнира не прибегая к редактированию конфигурационных файлов.

---

[1] Пьюривал, С. Основы разработки web-приложений / Сэмми Пьюривал; пер. с англ. А.Топлеева. – СПб.: Питер, 2015. – 272 с

– Поддержка практически всех популярных языков программирования: C, C++, Java, Pascal, Perl, Python и т.д. При компиляции системы автоматически выявляются установленные среды программирования и выполняется начальное конфигурирование. Добавление новых языков возможно и в уже работающую систему.

– «Безопасный» запуск тестируемых программ на выполнение (требует специальной поддержки со стороны ядра ОС). Тестируемая программа не может выполнить потенциально опасные операции, такие как создание новых процессов, открытие произвольного файла, создание сокета и т.п.

Также стоит отметить, что сегодня многие организации достаточно часто сталкиваются с необходимостью проведения того или иного тестирования. Например, психологическое тестирование, которое позволяет выявить "климат" и определить людей, которые являются источниками напряжения и мешают нормально работать другим. А ведь еще есть тесты по технике безопасности, профессиональные тесты для подтверждения права заниматься той или иной деятельностью, различные аттестации и т. п. Для этого очень удобно использовать компьютерное тестирование. Это позволяет ускорить проведение опросов и существенно облегчить обработку результатов.

Для обработки результатов наиболее подходит централизованная схема организации системы, т.е. клиент-сервер. Причем клиентов одновременно может работать несколько. Данная схема позволяет проводить тестирование одновременно для нескольких людей, например, целой учебной группы, и по окончании процедуры последним тестирующимся получить конечную статистику.

Наилучшим применением схемы клиент-сервер будет реализация ее как в локальной, так и в глобальной сетях. Так, в локальной сети удобно проводить тестирование на лабораторном занятии в компьютерном классе сразу всей группы. Вариант с глобальной сетью подойдет для тестирования знаний и навыков при дистанционном обучении, получившем широкое распространение в последнее время, а также для тестирования знаний обучающихся, которые по

некоторым причинам (например, по состоянию здоровья) не могут присутствовать.

Актуальность использования глобальной сети в качестве среды передачи данных подтверждает также ее повсеместное распространение и доступность.

Взаимодействующими сущностями, в порядке убывания иерархической значимости, являются:

1. Преподаватель (администратор);
2. Серверная часть системы;
3. Среда передачи данных;
4. Клиентская часть системы;
5. Пользователь (тестирующийся);

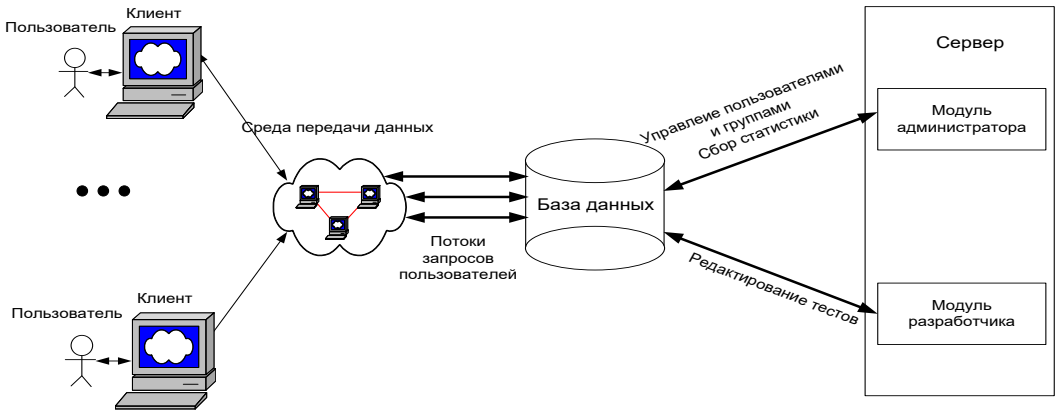


Рисунок 1 – Схема взаимодействия

Администратор же, в свою очередь, имеет возможность редактировать тестовые наборы, изменять, т.е. настраивать, процедуру тестирования, вносить изменения в список зарегистрированных пользователей, а также просматривать детальную статистику необходимого вида, получение которой и является окончательной целью.[30]

1.3 Выбор модели жизненного цикла

Для выбора модели ЖЦ конкретного проекта подходящей к условиям и систем Институтом качества программного обеспечения рекомендуется специальную процедуру использовать. Данная процедура из четырех таблиц вопро-

сов базируется на применении, прописана в положении «О внедрение автоматизированной системы в учебный процесс» от 02.02.2015 года. Примерные вопросы приведены в таблицах.

Таблица 5 - Выбор на основе характеристик требований модели жизненного цикла

№ п/п	Критерии категории требований	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Требования к проекту являются ли легко реализуемыми и определимыми ?	Да	Да	Да	Нет	Нет	Нет
2.	Требования могут ли быть в начале ЖЦ сформулированы?	Да	Да	Да	Да	Нет	Нет
3.	На протяжении ЖЦ часто ли будут изменяться требования?	Нет	Нет	Нет	Нет	Да	Да
4.	С целью их определения нужно ли демонстрировать требования?	Нет	Нет	Да	Нет	Да	Да
5.	Концепции программного средства требуется ли проверка системы?	Нет	Нет	Да	Нет	Да	Да
6.	Требования будут ли изменяться или уточняться в ЖЦ с ростом сложности системы (программного средства)?	Нет	Нет	Нет	Да	Да	Да
7.	На ранних этапах разработки необходимо реализовать основные требования?	Нет	Нет	Да	Да	Да	Да

Таблица 6 - Модель жизненного цикла выбор на основе характеристик разработчика

№ п/п	Критерии категории требований	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Предметная область проекта для большинства разработчиков являются ли проблемы новыми?	Нет	Нет	Нет	Нет	Да	Да
2.	Используемые в проекте для разработчиков, инструментальные средства для большинства являются ли новыми?	Да	Да	Нет	Нет	Нет	Да
3.	Роль участников ЖЦ проекта изменяются ли на протяжении?	Нет	Нет	Нет	Да	Да	Да
4.	Структура процесса для разработчиков более значимая является ли чем гибкость?	Да	Да	Нет	Да	Нет	Нет
5.	Человеческий ресурс проекта важна ли легкость распределения?	Да	Да	Да	Да	Нет	Нет
6.	Команда разработчиков в стадии разработки оценки, проверки, приемлет ли?	Да	Да	Нет	Да	Да	Да

Таблицы 2 – 5 категорий классификации представляет одну из проектов. По определенному критерию категории каждый из вопросов (строка в таблице) классификации предназначен для анализируемого проекта. Модель ЖЦ соответствуют столбцам данных таблиц обобщенным, фактически представляющим стратегии разработки ПС. Независимая RAD-модель, при этом под RAD-моделью подразумевается жизненного цикла не встроенная в другие модели. из Следующая последовательность рассматриваемая процедура состоит из шагов.

Таблица 7 – Модель жизненного цикла выбор на основе характеристик пользователей

№ п/п	Критерии категории требований	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Будет ли ограничено в ЖЦ разработки присутствие пользователей?	Да	Да	Нет	Да	Нет	Да
2.	Состояние программного продукта (подсистемы) будут ли пользователи оценивать текущее в процессе разработки?	Нет	Нет	Нет	Да	Да	Да
3.	ЖЦ разработки во влечены во все фазы будут ли пользователи?	Нет	Нет	Да	Нет	Да	Нет
4.	Ход выполнения проекта будет ли заказчик отслеживать?	Нет	Нет	Нет	Нет	Да	Да

1-й шаг. Проанализировать отличительные черты проекта по критериям категорий, представленным в виде вопросов.

2-й шаг. Ответить на вопросы по анализируемому проекту, отметив слова «да» или «нет» в соответствующих строках таблиц 1 – 4. Если слов «да» или «нет» в строке несколько, необходимо отметить все из них (все «да» или все «нет»).

В качестве примера в таблице 1 выделены варианты ответов для проекта разработки сложного и критичного программного средства, требования к которому заранее не известны и будут уточняться по ходу разработки.

3-й шаг. Расположить по степени важности категории (таблицы) и/или критерии, относящиеся к каждой категории (вопросы внутри таблиц), относительно проекта, для которого выбирается модель ЖЦ.

4-й шаг. Выбрать из моделей (см. таблицы 1 – 4) ту модель, которая соответствует столбцу с наибольшим количеством отмеченных ответов с учетом их степени важности (с наибольшим количеством отмеченных ответов в верхней

части приоритетных таблиц). Выбранная модель ЖЦ является наиболее приемлемой для анализируемого проекта.

Таблица 8 - Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

№ п/п	Критерии категории типов проекта и рисков	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Продукт нового для организации направления разрабатывается ли в проекте?	Нет	Нет	Нет	Да	Да	Да
2.	Проект будет ли являться расширением существующей системы?	Да	Да	Да	Да	Нет	Нет
3.	Крупно- или среднемасштабным будет ли проект?	Нет	Нет	Да	Да	Да	Да
4.	Длительная эксплуатация ожидается ли продукта?	Да	Да	Да	Да	Нет	Да
5.	Проекту продукта необходим ли высокий уровень надежности?	Нет	Да	Нет	Да	Нет	Да
6.	В течение ЖЦ предполагается ли эволюция продукта проекта?	Нет	Нет	Нет	Да	Да	Да
7.	На этапе сопровождения велика ли вероятность изменения системы (продукта)?	Нет	Нет	Нет	Да	Да	Да
8.	Сжатым является ли график?	Нет	Нет	Да	Да	Да	Да
9.	Использование компонентов предполагается ли повторное?	Нет	Нет	Да	Да	Да	Да
10.	Ресурсы являются ли достаточными (время, деньги, инструменты, персонал)?	Нет	Нет	Нет	Нет	Да	Да

1.4 Обоснование модели жизненного цикла для разрабатываемого программного средства

«В настоящее время имеется несколько методологий разработки программного обеспечения, которые можно рекомендовать при использовании спиральной модели жизненного цикла. Наиболее известными из них являются методология быстрой разработки приложений (Rapid Application Development, RAD) и экстремальное программирование (eXtreme Programming)».[9]

«Методология RAD. На начальном этапе существования компьютерных информационных систем их разработка велась на традиционных языках программирования и подразумевала, как правило, ручной набор текстов программ. Однако по мере возрастания сложности разрабатываемых систем и увеличения запросов пользователей потребовались новые средства, обеспечивающие значительное сокращение сроков разработки. Это послужило предпосылкой к созданию инструментальных средств для быстрой разработки приложений. Развитие этого направления привело к появлению на рынке разработки программного обеспечения средств автоматизации практически всех стадий жизненного цикла».[15]

«Под RAD-разработкой обычно понимается процесс разработки, содержащий 3 элемента:

– небольшую команду программистов (до 10 человек);

– короткий, но тщательно проработанный производственный график (от 2 до 6 месяцев);

– повторяющийся цикл, при котором разработчики по мере того, как приложение начинает обретать форму, реализуют в продукте требования, полученные через взаимодействие с заказчиком».[15]

«Помимо особенностей, характерных для спиральной модели жизненного цикла, методология RAD подразумевает использование на каждой итерации:

– Case-средств для формирования и анализа требований, проектирования системы, автоматической генерации кода программ и структуры БД, а также автоматического тестирования программного обеспечения;

– инструментальных средств, обеспечивающих визуальную разработку (программирование) системы. Среда разработки приложений позволяет без написания кода программы создавать («рисовать») сложные графические интерфейсы пользователя, состав и структуру БД, запросы к БД, а также связывать данные с элементами интерфейса (переключателями, полями ввода, таблицами и т. д.);

– инструментальных средств, поддерживающих объектно-ориентированный подход. Эти средства позволяют создать описание предметной области в виде совокупности объектов – сущностей реального мира, характеризуемых свойствами (атрибутами) и поведением (методами);

– инструментальных средств, обеспечивающих событийное программирование. Каждый объект, входящий в состав приложения, может генерировать события и реагировать на события, генерируемые другими объектами;

– шаблонов и библиотек готовых решений как собственной разработки, так и сторонних производителей».[9]

«Условия применения методологии RAD:

– применима для относительно небольших проектов, разрабатываемых под конкретного заказчика;

– неприменима для построения сложных расчетных программ, операционных систем или программ управления космическими кораблями, т.е. программ, требующих написания большого объема (сотни тысяч строк) уникального кода;

– неприменима для разработки приложений, в которых отсутствует ярко выраженная интерфейсная часть, наглядно определяющая логику работы системы (например, приложения реального времени);

– неприменима для разработки приложений, от которых зависит безопасность людей (например, управление самолетом или атомной электростанцией), так как итеративный подход предполагает, что первые несколько версий, скорее всего не будут полностью работоспособны, что в данном случае исключается».[15]

Таким образом, рассмотренная выше методология, направлена на сокращение сроков и расходов по разработке приложений с одновременным повышением качества результатов работы. Главное достоинство этой методологии заключается в наиболее полном и точном удовлетворении требований организации за счет обеспечения своевременной обратной связи».

2 АНАЛИЗ МЕТОДОВ ПРОВЕДЕНИЯ ТЕСТИРОВАНИЯ

2.1 Методы решения проблем тестирования

«Олимпиада по программированию – интеллектуальное соревнование по решению комплекта из задач на компьютере, для решения которых необходимо придумать и реализовать алгоритм, написав программный код на одном из языков программирования. Задача считается решённой, если составленная по коду участника программа выдает верный результат на всем наборе подготовленных тестов, которые заранее неизвестны никому кроме составителей».[22]

Для соревнований используются турнирные системы проведения для проверки решений участников, а также обеспечения взаимодействия клиентов с сервером. Сервер – это программа, запускаемая на отдельном ПК, и выполняющая определенные задачи. Для работы сервера к которому будет обращаться клиент обычно выделяется порт, для пользования его ресурсами. Клиент – программа, выполнить какую-либо задачу позволяющая запрашивать и вернуть полученные данные клиенту у сервера (внести, изменить, удалить, найти информацию).

На сервере производится создание соревнования, предварительная регистрация участников и добавление заданий с заранее заготовленными наборами тестов для них. Участники должны написать код программы, решающий конкретную задачу, и отослать на сервер, где решение скомпилируется и пройдёт проверку на имеющемся наборе тестовых данных.

Участник получает сообщение по результатам прохождения тестов о правильности своего решения. Набор тестов – набор данных, которые используются для проверки решения на правильность. Считается, что решение проходит тестирование, если полученная программа успешно завершается, не превысив ограничений на ресурсы, и выдает верный ответ. Корректность решения проверяет специальная программа – чекер (от англ. Checker), определяющая эквивалентны ли фактический (ответ участника) и ожидаемый (правильный ответ) результат работы программы. В первую очередь, тестирующая система

призвана автоматизировать процесс проверки решения, присланного участником на обрабатывающий эту информацию сервер, что, безусловно, ускоряет подведение итогов и сохраняет динамическую составляющую состязания. Вследствие того, что больше не будет требоваться ручная проверка – участник соревнования может сразу узнать свой результат и сравнить его с результатами других участников.

Во-вторых, система позволяет отстранить людей от проверки решений, возлагая всю ответственность на машину, что исключает влияние человеческого фактора на вердикт (человек может ошибиться или выставить неверный результат намеренно, для личной выгоды). Ввиду того, что количество человек, участвующих в соревновании, может доходить до нескольких тысяч – важно наличие постоянной обратной связи, возможности задать вопрос и быстро получить ответ. Однако внедрение системы может повлечь за собой нежелательные последствия.

Например, в системе может возникнуть ошибка, из-за которой вердикт будет неверным или баллы будут вычислены некорректно. Иногда проблемой является её уязвимость, участники могут воспользоваться этим и обмануть систему, что позволит им изменить ход соревнования. В настоящее время тестирующие системы проектируют таким образом, чтобы избежать утечек по части безопасности. Следующее последствие связано с тем, что пользоваться тестирующей системой не всегда легко.[8]

Соответственно, необходимо ознакомление, разнообразная помощь. Важно отметить, что проверка на корректность и качество кода, необходимая в профессиональном программировании, в последнее время на олимпиадах практически не производится. Объясняется это тем, что количество предлагаемых на одном туре задач и их сложность со временем возросли и во время олимпиады важнее определить не степень профессиональной пригодности участника как программиста, а его способность решать те или иные задачи. [22]

Преимущества использования тестирующей системы Одно из технических требований при использовании системы для автоматической проверки –

задачи должны формулироваться очень строго, с указанием всех возможных ограничений (по времени и ресурсам). Также при использовании системы от пользователей требуется очень жесткое соблюдение форматов входных и выходных данных (иначе автоматическая проверка невозможна). Впрочем, это скорее одно из достоинств использования системы: школьники привыкают точно следовать требованиям технического задания (а это также очень ценный навык, и ценный опять же не только для программиста).

Подготовка тестов – сложный процесс, требующий высокой квалификации, если учитывать требование, что в тестовом наборе ни в коем случае не должно быть ошибок. Однако однажды подготовленные наборы тестов могут впоследствии использоваться сколько угодно раз.

Таким образом, для эффективной работы пользователя от системы определены следующие потребительские свойства:

– доступность информации в реальном времени;

– наглядность информации, простота интерфейса;

– надежность сетевого взаимодействия;

– безопасность персональных данных;

– удобство использования;

– отсутствие возможности использования уязвимостей ПО;

– системы подсказок и обучения.

Система поддерживает различные режимы работы; может быть дополнена другими видами тестирования. Скорость Система работает в режиме реального времени; сервер быстро обрабатывает запросы и отправляет ответные данные. Простота легко устанавливается и настраивается; интерфейс клиентской части интуитивно понятен для среднестатистического пользователя.[29]

Для решения проблемы проверки решения задания разными учениками используются методы тестирования кода.

Методы тестирования можно разделить по так называемым «уровням».

а) Интеграционное тестирование – тестируются интерфейсы между компонентами, подсистемами или системами. При наличии резерва времени на

данной стадии тестирование ведѐтся итерационно, с постепенным подключением последующих подсистем.

б) Системное тестирование – тестируется интегрированная система на еѐ соответствие требованиям. Для решения данной задачи будет использоваться модульное тестирование так как этот метод обладает такими плюсами как:

в) Поощрение изменений –модульное тестирование позже позволяет программистам проводить рефакторинг, будучи уверенными, что модуль по-прежнему работает корректно (регрессионное тестирование). Это поощряет программистов к изменениям кода, поскольку достаточно легко проверить, что код работает и после изменений. [18]

г) Упрощение интеграции– модульное тестирование помогает устранить сомнения по поводу отдельных модулей и может быть использовано для подхода к тестированию «снизу-вверх»: сначала тестируя отдельные части программы, а затем программу в целом.

д) Документирование кода– модульные тесты можно рассматривать как «живой документ» для тестируемого класса. Разработчики, которые не знают, как использовать данный класс, могут использовать юнит-тест в качестве примера.

е) Отделение интерфейса от реализации– поскольку некоторые классы могут использовать другие классы, тестирование отдельного класса часто распространяется на связанные с ним. Например, класс пользуется базой данных; в ходе написания теста программист обнаруживает, что тесту приходится взаимодействовать с базой. Это ошибка, поскольку тест не должен выходить за границу класса. В результате разработчик абстрагируется от соединения с базой данных и реализует этот интерфейс, используя свой собственный mock-объект. Это приводит к менее связанному коду, минимизируя зависимости в системе. В системе автоматической проверки решений будет использоваться модульное тестирование кода.[10]

ж) Модульное тестирование – тестируется минимально возможный для тестирования компонент, например, отдельный класс или функция. Часто мо-

дульное тестирование осуществляется разработчиками программного обеспечения.

Существует несколько фреймворков для автоматического модульного тестирования:

- Mocha.
- Chai («assertion library», используется совместно с тестовым framework'ом).
- Sinon.JS (библиотека для создания mock'ов, stub'ов, spy'ев, используется совместно с тестовым framework'ом).
- Karma runner (от создателей Angular.JS, «test runner» – организует среду выполнения тестов).
- Qunit (от создателей jQuery).
- JsUnit (больше не поддерживается создателями).
- Jasmine (рекомендован создателями jsUnit).
- D.O.H.

Таким образом рассмотрены основные проблемы обучения программированию, выбраны основные средства реализации и проведен обзор существующих решений данной проблемы. Необходимо спроектировать и разработать систему автоматической проверки заданий. Данная система должна удовлетворять следующим требованиям:

1. Обладать высоким коэффициентом надежности.
2. Корректно проверять написанный код учеников.
3. Обеспечивать максимальное быстродействие при проверке заданий.

2.2 Подготовка задачи Tests

Задачи являются самыми сложными для подготовки. При описании подготовки задач будет предполагаться, что турнир настраивается
в альтернативной раскладке файлов, то есть все файлы, относящиеся к задаче, размещаются в одном каталоге.

Настройка основных параметров задачи

Поскольку задача на написание тестов предполагает запуск программ на тестовых данных аналогично стандартной задаче, в разделе конфигурации задачи должны быть установлены параметры, относящиеся к тестированию задач. Например:

```
# Идентификационные параметры задачи
id = …
short_name = …
long_name = …
# Основные параметры задачи
test_sfx = «.dat»  # Суффикс имен файлов с тестовыми данными
use_corr  # Проверка использует файлы с правильными ответами
corr_dir = «%Ps»  # Каталог с файлами с правильными ответами – значение не важно
corr_sfx = «.ans»  # Суффикс имен файлов с правильным ответом
use_stdin  # Программа считывает результат со стандартного потока ввода
use_stdout  # Программа выводит результат на стандартный поток вывода
standard_checker = «cmp_int» # Проверка – сравнение двух целых чисел
time_limit = 1  # Ограничение времени ЦП – 1 секунда
real_time_limit = 5 # Ограничение реального времени – 5 секунд
max_stack_size = 8M  # Ограничение размера стека – 8 мегабайт
max_vm_size = 64M  # Ограничение общего размера вирт. Памяти – 64 мегабайта
```

Идентификационные параметры задачи – это идентификатор задачи id, короткое название short_name, полное название long_name и, возможно, внутреннее название задачи internal_name.

Значение параметра corr_dir несущественно, так как предполагается что турнир настраивается в альтернативной раскладке файлов. Тем не менее, параметр corr_dir должен быть установлен в некоторое непустое значение.

Настройка дополнительных параметров для задач на разработку тестов

Для задачи на написание тестов должны быть установлены следующие параметры задачи.

type = «tests»

binary

enable_language = «application/x-gzip»

enable_language = «application/x-compress»

enable_language = «application/x-bzip2»

enable_language = «application/x-tar»

enable_language = «application/zip»

Написание проверяющей программы для тестов

Следующий шаг – написание программы для проверки корректности тестов. Эта программа получает два аргумента командной строки: имя файла с тестовыми данными и имя файла с правильным результатом работы. Программа проверки корректности тестов может, в принципе, модифицировать и первый, и второй файл. Для последующего тестирования будут передаваться уже модифицированные файлы.

Рассмотрим программу для проверки корректности тестов для задачи сложения двух целых чисел. На стандартном потоке ввода задаются два целых числа в диапазоне [-32768;32767]. На стандартный поток вывода необходимо напечатать сумму этих чисел.

Программу проверки корректности тестов можно писать на любом языке (в том числе и скриптовом). В нашем случае эта программа будет написана на языке Си с использованием библиотеки libchecker.

```
#include «checkutils.h»
#include <limits.h>
int
main(int argc, char **argv)
{
  int x, y, z;
```

```
    /* проверить, что передано правильное число аргументов командной строки */
    if (argc != 3) fatal_CF(«wrong number of arguments»);
    /* открыть файл с тестовыми данными и прочитать два числа */
    checker_in_open(argv[1]);
    checker_read_int_ex(f_arr[0], fatal_PE, «x», 1, &x);
    checker_read_int_ex(f_arr[0], fatal_PE, «y», 1, &y);
    checker_eof(f_arr[0], fatal_PE, «test input»);
    checker_in_close();
    /* проверить ограничения на входные данные */
    if (x < SHRT_MIN || x > SHRT_MAX)
      fatal_PE(«first value is out of range»);
    if (y < SHRT_MIN || y > SHRT_MAX)
      fatal_PE(«second value is out of range»);
    /* открыть файл с правильным ответом и прочитать число */
    checker_out_open(argv[2]);
    checker_read_int_ex(f_arr[1], fatal_PE, «z», 1, &z);
    checker_eof(f_arr[1], fatal_PE, «test answer»);
    checker_out_close();
    /* проверить правильность ответа */
    if (z != x + y)
      fatal_WA(«wrong answer»);
    return 0;
  }
```

3 АРХИТЕКТУРНЫЙ ПРОЕКТ

Рассмотрены основные проблемы обучения программированию, выбраны основные средства реализации и проведен обзор существующих решений данной проблемы. Необходимо спроектировать и разработать систему автоматической проверки заданий. Данная система должна удовлетворять следующим требованиям:

1) Обладать высоким коэффициентом надежности

2) Корректно проверять написанный код учеников

3) Обеспечивать максимальное быстродействие при проверке заданий

Требуется автоматизировать процесс проведения Всероссийской олимпиады школьников по дисциплине «Информатика». Описание предметной области диаграммой IDEF0, воспользуемся составленной в программе AllFusion Process Modeler R7.

Контекстную диаграмму системы определим таким образом представленная на рисунке 2.

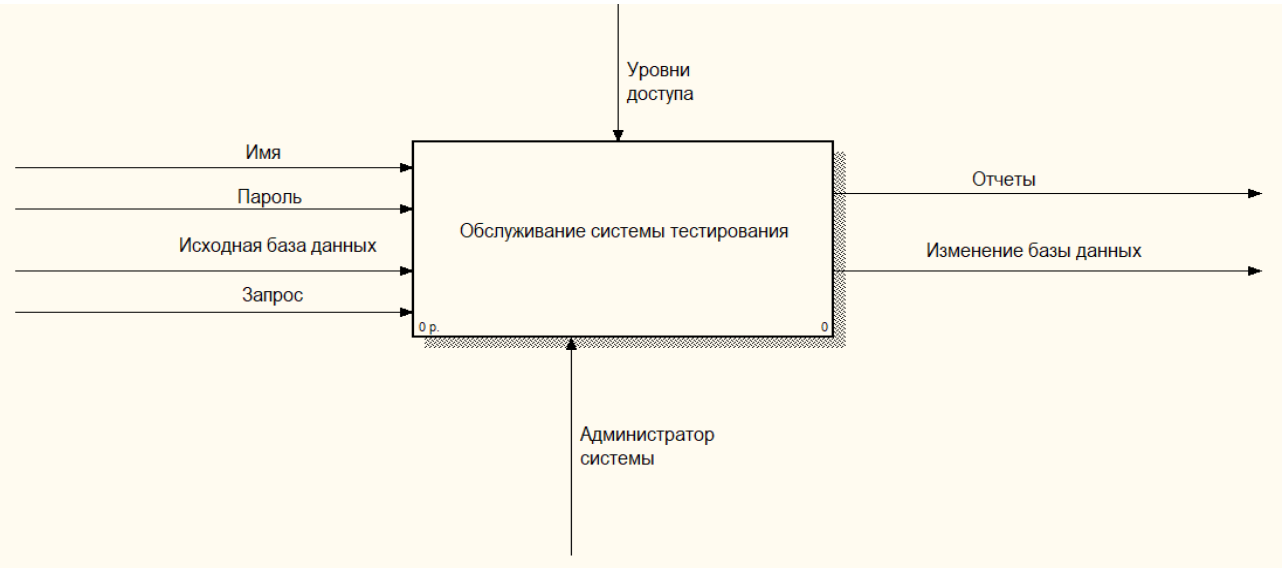


Рисунок 2 – Контекстная диаграмма системы

Декомпозицию контекстной диаграммы проведем, описав последовательность обслуживания пользователя:

- Определение уровня доступа в систему.
- Выбор подсистемы.
- Обращение к подсистеме.
- Изменение БД

Таким образом, получим диаграмму, представленную на рисунке 3.

Декомпозицию контекстной диаграммы закончив, переходим к следующему уровню декомпозиции диаграммы. Обычно модели возвращаются к родительским диаграммам при рассмотрении третьего и более нижних уровней и корректируют их.

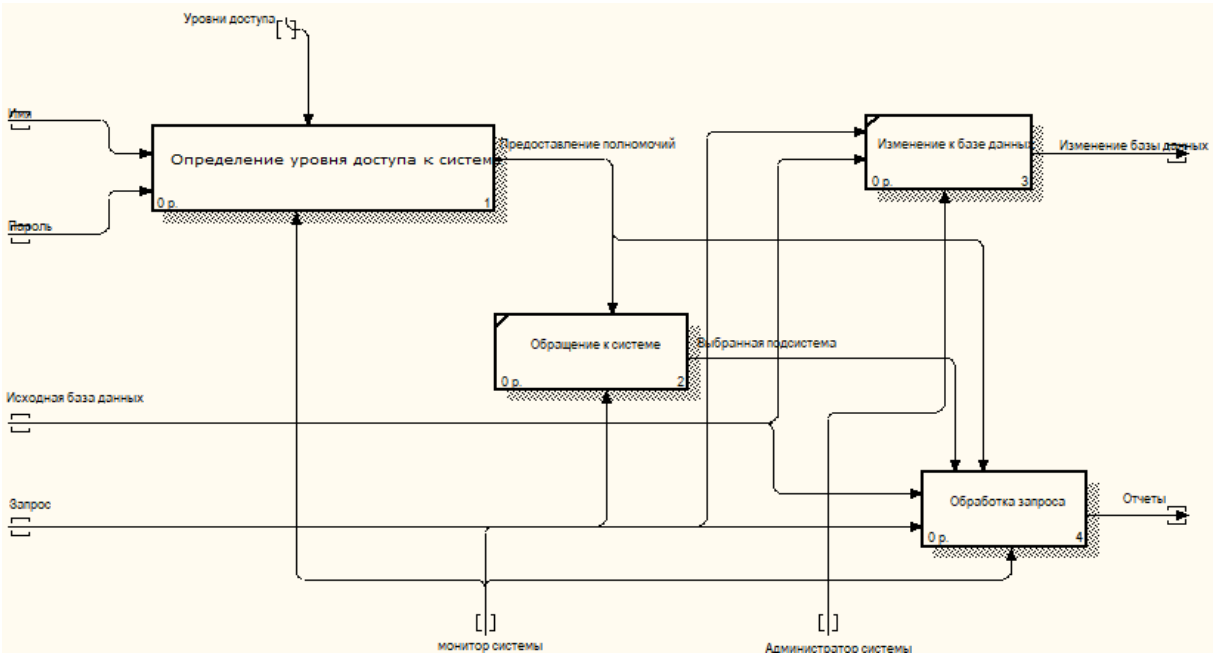


Рисунок 3 – «Обслуживание системы» декомпозиция работы

Все блоки полученной диаграммы декомпозируем последовательно. При определении уровня доступа в систему первым этапом является определение категории пользователя. По имени определяя его категорию осуществляется поиск в базе пользователей. Предоставляемые пользователю системы выясняются полномочия, согласно определенной категории. Проверяя имя и пароль доступа далее проводится процедура доступа в систему. Для пользователя формируется набор разрешенных действий, объединяя информацию о полномочиях

и уровне доступа в систему. Таким образом, определение уровня доступа в систему будет изображено на рисунке 4.

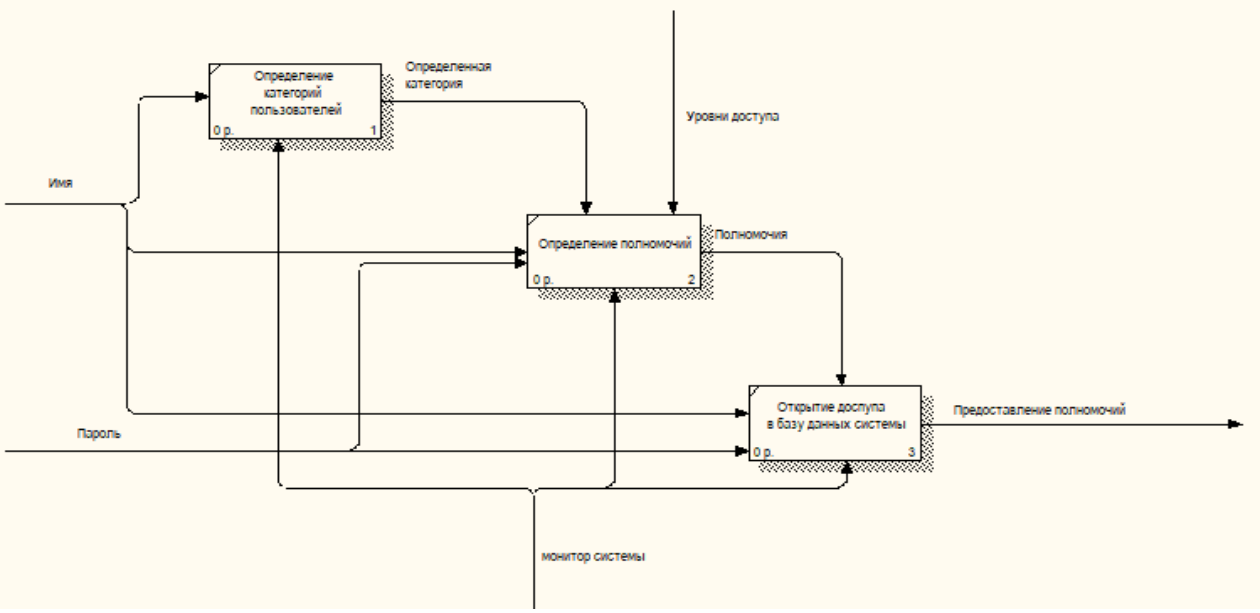


Рисунок 4 – «Определение уровня доступа в систему» декомпозиция работы

Выбирая подсистему, которая будет обрабатывать запрос после прохождения процедуры доступа в систему монитор анализирует запрос. «Обращение к подсистеме» декомпозиция работы не отвечает цели и точке зрения модели. Внутренние алгоритмы ее работы пользователя системы не интересуют. Поэтому декомпозиция обращения к подсистеме только усложнит модель в данном случае ему важно, что выбор подсистемы будет произведен автоматически, без его вмешательства.

«Обработка запроса пользователя» декомпозируем работу, определения категорий и полномочий пользователей выполняемой подсистемой обработки запросов. Необходимо открыть БД (подключиться к ней) перед осуществлением поиска ответа на запрос. В общем случае БД может потребовать установление соединения с ней находясь на удаленном сервере. Определим последовательность работ:

- БД открытие,
- запрос выполнения,

- отчет генерации.

Выполнить запрос и сгенерировать отчеты для пользователя после открытия БД необходимо сообщить системе об установлении соединения с БД (рисунок 5).

На «Выполнение запроса» необходимо отметить, включается работа различных подсистем. Например, подсистема профессиональных тестов если запрос включает в себя тестирование, то его будет исполнять. На этапе выполнения запроса может потребоваться изменение содержимого БД, например при составлении экспертных оценок. На диаграмме необходимо поэтому предусмотреть такую возможность.

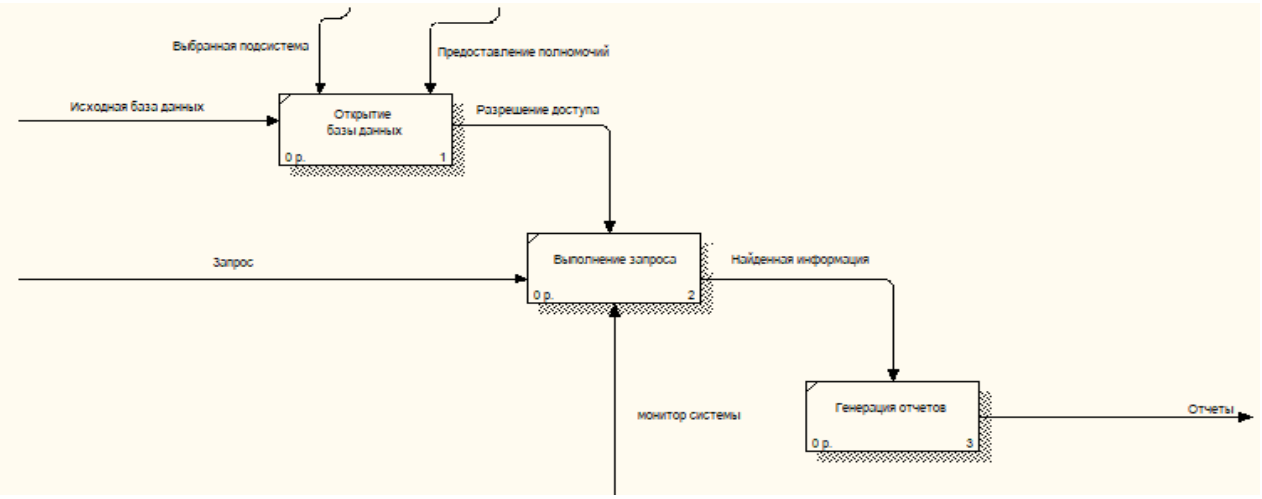


Рисунок 5 – Декомпозиция работы «Обработка запроса пользователя»

«По каким правилам происходит генерация отчетов, при анализе полученной диаграммы возникает вопрос? Шаблоны должны соответствовать запросам и должны быть заранее определены, необходимо наличие заранее сформированных шаблонов, по которым будет производиться выборка из БД. Пользователю должна быть предоставлена возможность выбора формы отчета».[6]

Были выявлены следующие проблемы в ходе изучения предметной области:

– в процессе обучения на прохождение каждого этапа затрачивается слишком много времени;

– тесты не всегда осуществляется с помощью ПО и технических средств тестирование и проверка, а зачастую происходит вручную;

– оценивании знаний школьников играет роль людской фактор, при подведении результатов, что зачастую не всегда сказывается благоприятным образом на знании учащегося.

Данные проблемы могут быть решены, анализ научной литературы показывает, что с использованием автоматизированных обучающих систем.

«С позиций современной дидактики введение информационной среды и программного обеспечения внесло огромное количество новых возможностей во все области процесса обучения. Автоматизированные обучающие системы представляют собой комплексы научно-методической, учебной и организационной поддержки процесса обучения, проводимого на базе компьютерных, или, как их также называют, информационных технологий. За счет своего быстродействия и больших резервов памяти они позволяют реализовывать различные варианты сред для программированного и проблемного обучения, строить различные варианты диалоговых режимов обучения, когда так или иначе ответ учащегося реально влияет на ход дальнейшего обучения. Компьютерные технологии представляют собой принципиально новые средства обучения».[4]

В общем случае педагог получает доступ к компьютерным средствам, информационной среде и программным продуктам, предназначенным для обеспечения преподавательской деятельности. Вследствие этого современный педагог с неизбежностью должен осваивать новые образовательные подходы, опирающиеся на средства и методы индивидуального компьютерного обучения. Все эти средства образуют комплексы автоматизированных обучающих систем.

В рамках автоматизированных обучающих систем на сегодняшний день решается ряд задач обучения. В первую группу можно отнести задачи проверки уровня знаний, умений и навыков учащихся до и после обучения, их

индивидуальных способностей, склонностей и мотиваций. Для таких проверок обычно используют соответствующие системы тесты и экзаменационные вопросы.

Техническое обеспечение автоматизированных обучающих систем основано на локальных компьютерных сетях, включающих автоматизированные рабочие места (АРМ) учащихся, преподавателя и линии связи между ними.

«В настоящее время разработано большое число электронных учебных материалов, в качестве которых выступают электронные учебники, электронные учебные пособия, автоматизированные обучающие системы и т.п. Существующие электронные учебные материалы решают те или иные задачи обучения с большей или меньшей эффективностью, которая определяется, прежде всего, степенью управляемости обучаемым в процессе обучения. В условиях нарастающего интереса, к созданию различных вариантов электронно-методических материалов возникает необходимость в классификации этих материалов с целью оценки их различия и определения области применения. Уже существует ряд классификаций обучающих систем по различным их свойствам. Однако нет классификации, отражающей управляемость обучаемого системой, что при расширяющемся использовании электронных учебных материалов, является важным на данный момент».[4]

3.1 Диаграмма вариантов использования

Диаграмма вариантов использования, которая описывает функциональное назначение системы с позиций различных групп пользователей, другими словами, то, что система будет делать в процессе своего функционирования в результате действий различных пользователей.

Диаграмма вариантов использования является исходным концептуальным представлением или концептуальной моделью системы в процессе ее проектирования и разработки. Диаграммы вариантов использования для системы в целом с позиций администратора системы, пользователя. Любой пользователь си-

стемы, должен быть зарегистрированным пользователем, поэтому для работы с системой должны пройти авторизацию.

Пользователь администратор имеет доступ ко всем данным системы и имеет право на их добавление, удаление, обновление. Администратор системы может параметры поиска и получать отчетную информацию, например, о зарегистрированных пользователях и их данных.

Рисунок 6 – Диаграмма вариантов использования «Администратор»

При прохождении олимпиады пользователь (обучающийся) при выполнении задания просматривает свой программный код загруженный на проверку может удалить его, а также внести изменения до улучшения результата.

Результаты проверяются в ранее заготовленных тестах сохранённые в базе данных (рисунок 7).

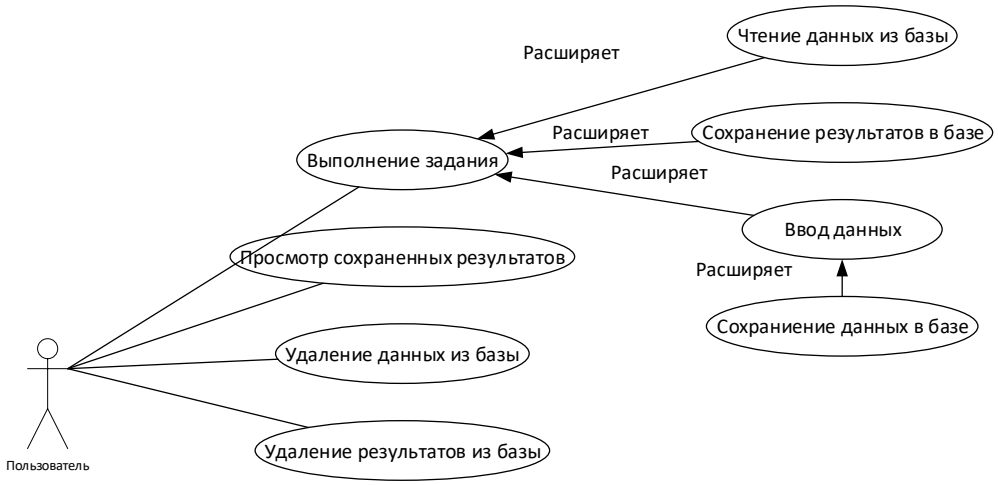


Рисунок 7 – Диаграмма вариантов использования «Пользователь»

3.2 Диаграмма последовательности

При авторизации в системе тестирования пользователь вводит логин и пароль, входит в программу. После авторизации пользователь попадает в личный кабинет, где ему отображается список назначенных тестов. Если тестирование имеет ограниченной срок сдачи, то в интерфейсе отображается дата, до которой необходимо пройти тестирование.

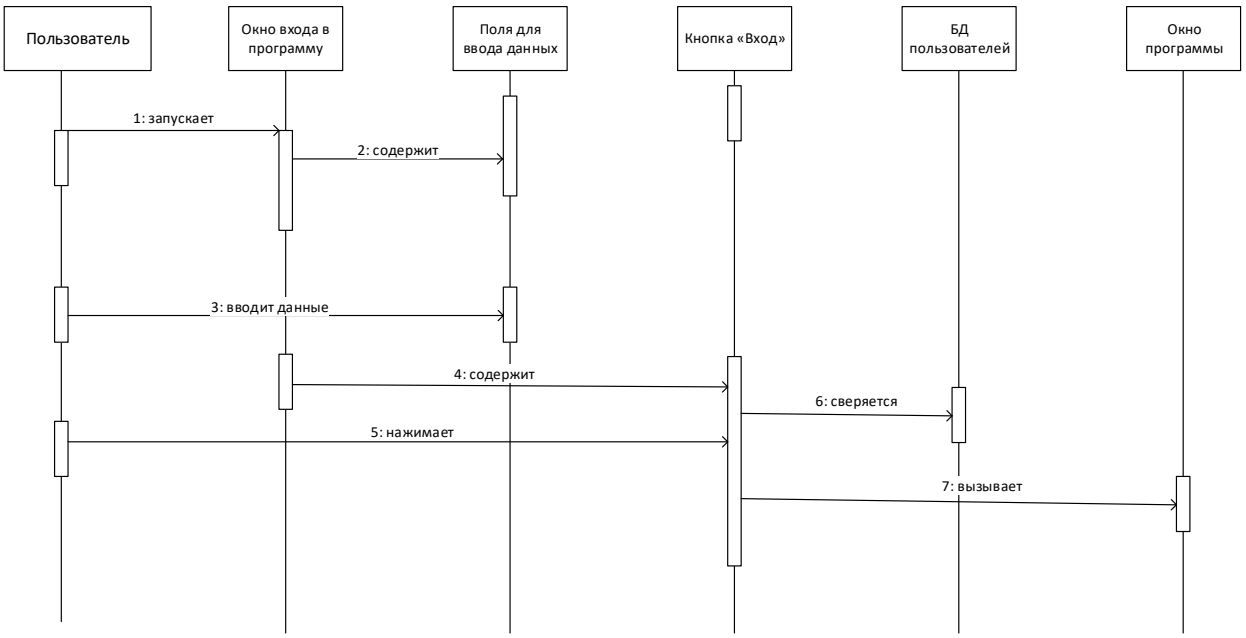


Рисунок 8 – Диаграмма вариантов последовательности «Авторизация пользователя»

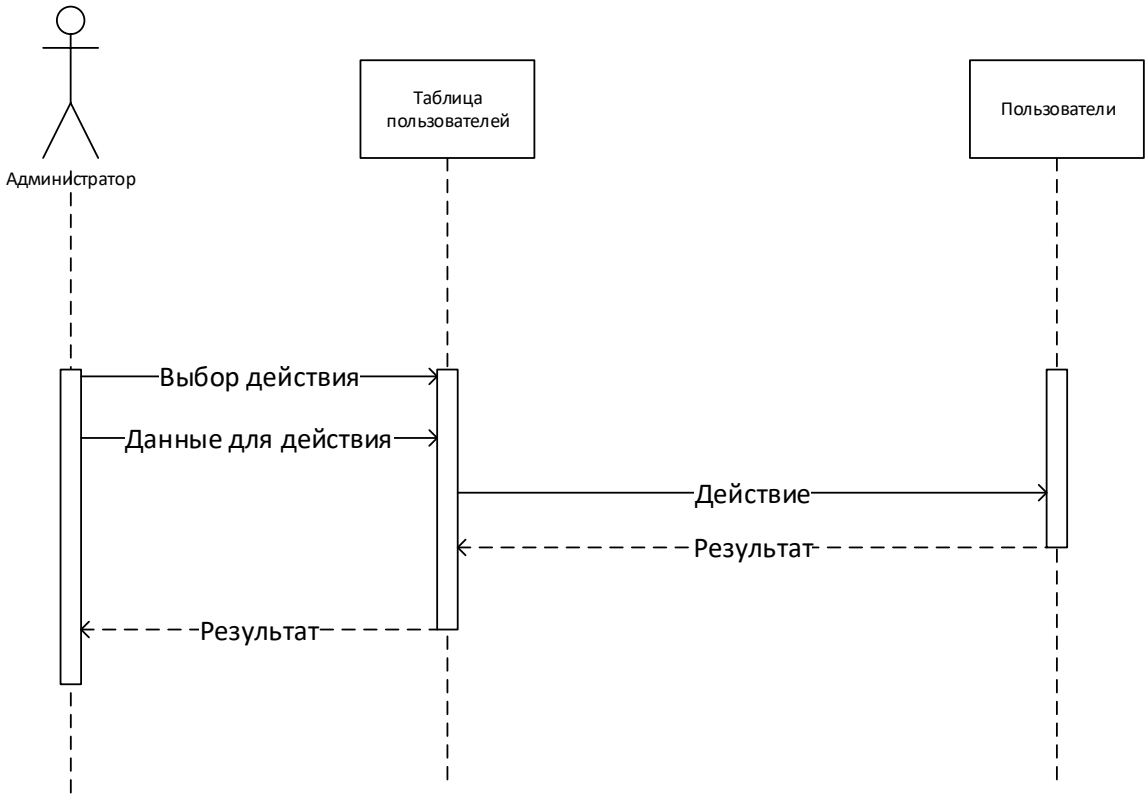


Рисунок 9 – Диаграмма вариантов последовательности «Администратор»

3.3 Диаграмма состояний подсистемы пользователь и администратор

Диаграмма состояний описывает возможные последовательности состояний и переходов, которые в совокупности характеризуют поведение системы в течение жизненного цикла. Диаграмма состояний представляет динамическое поведение сущностей, на основе спецификации их реакции на восприятие некоторых конкретных событий. Диаграммы состояний чаще всего используются для описания поведения отдельных систем и подсистем. Они также могут быть применены для спецификации функциональности экземпляров отдельных классов, т. е. для моделирования всех возможных изменений состояний конкретных объектов.

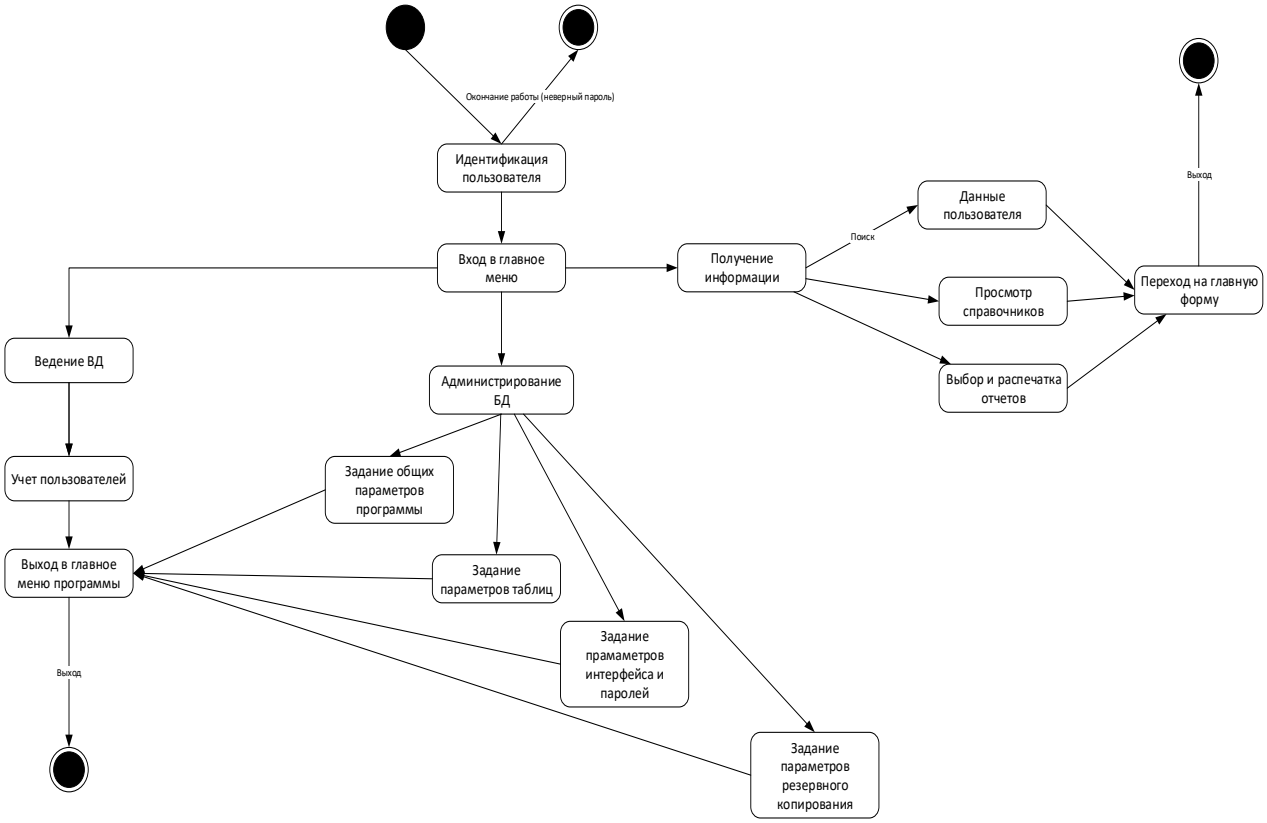


Рисунок 10 – Диаграмма вариантов состояний «Идентификация пользователя»

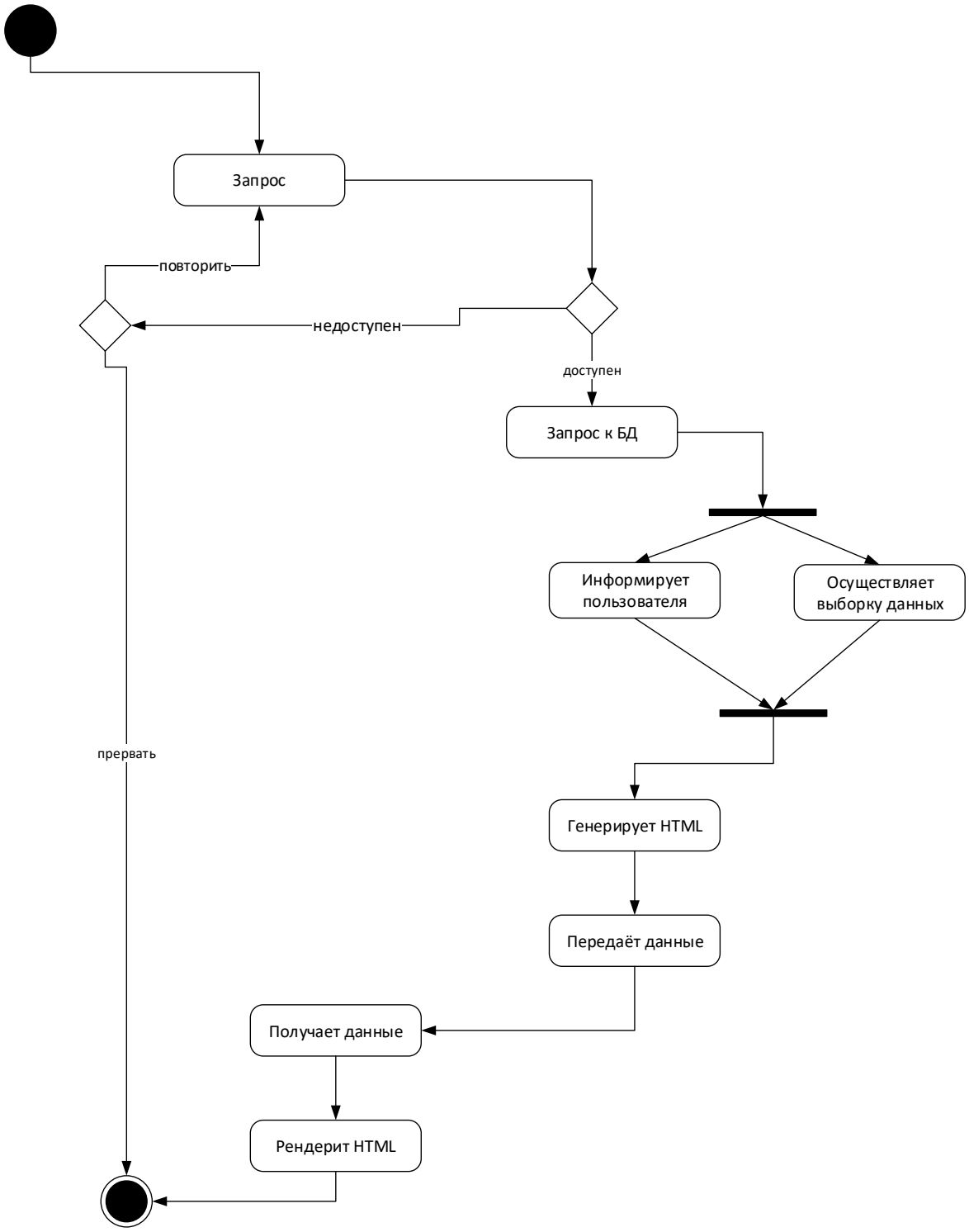


Рисунок 11 – Диаграмма вариантов состояний «Пользователь, Веб-сервер, Сервер БД»

4 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПОДСИСТЕМЫ

4.1 Разработка подсистемы ejudge

В данном алгоритме рассмотрим, как работает приложение в целом. Сервер принимает запросы от пользователей на котором находится приложение. После запуска сервера у нас подгружаются все библиотеки. Приложение ждет запросы от пользователей после предварительной подготовки. Как только запускается запросы и генерация страниц приходит запрос выполнения требуемых действий. Далее анализ ошибок в случае обнаружения неполадок сообщается администратору сервера и происходит проверка работоспособности системы.

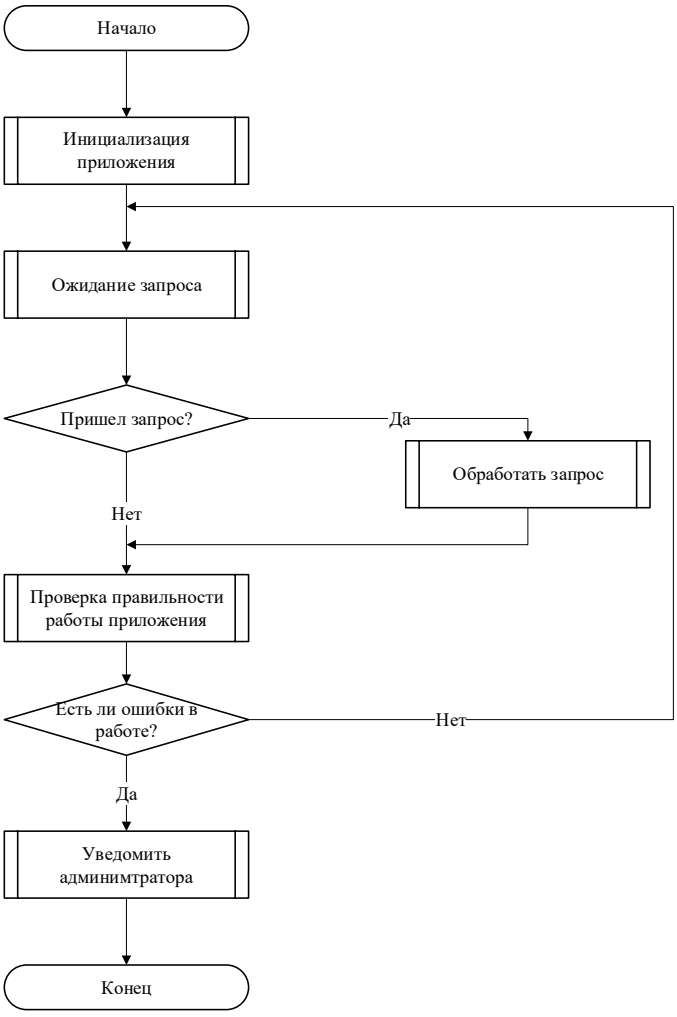


Рисунок 12 – Общий алгоритм приложения

В данном алгоритме рассмотрим, как система обрабатывает запросы. В запросе хранится путь нужной страницы, действия и дополнительные параметры. Все эти данные разбиваем на части для удобной работы с ними. Для работы с системой необходима авторизация пользователя. Если таковая отсутствует, то переадресуем пользователя на страницу авторизации рисунок 13.

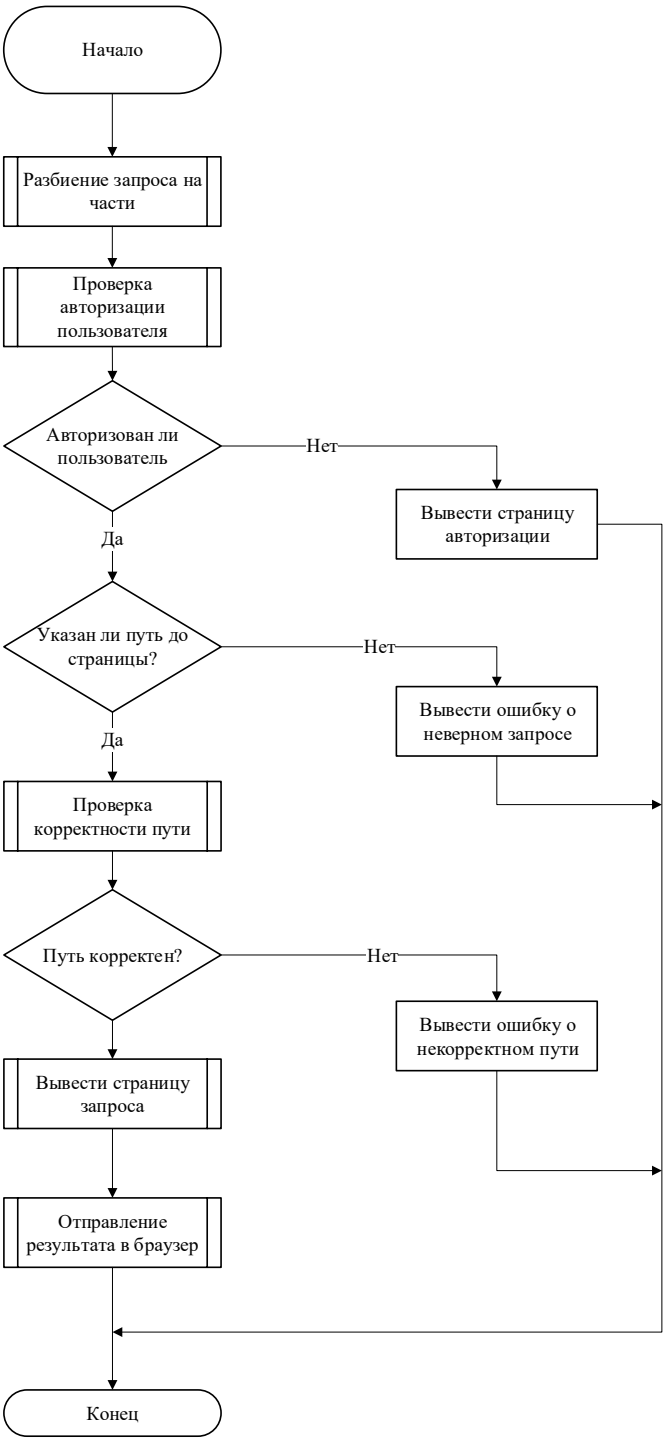


Рисунок 13 – Алгоритм обработки запроса

Далее проверяем, указан ли путь нужной страницы, если нет, то выводим сообщение о неверном запросе. Если же путь присутствует, то проверяем его правильность, если такого пути не существует, мы выводим ошибку, что некорректный путь.

Рассмотрим, как подсистема выдаёт страницы по запросам. Чтобы вывести страницу пользователю требуется прочитать действия и параметры запроса. Далее идем по списку действий.

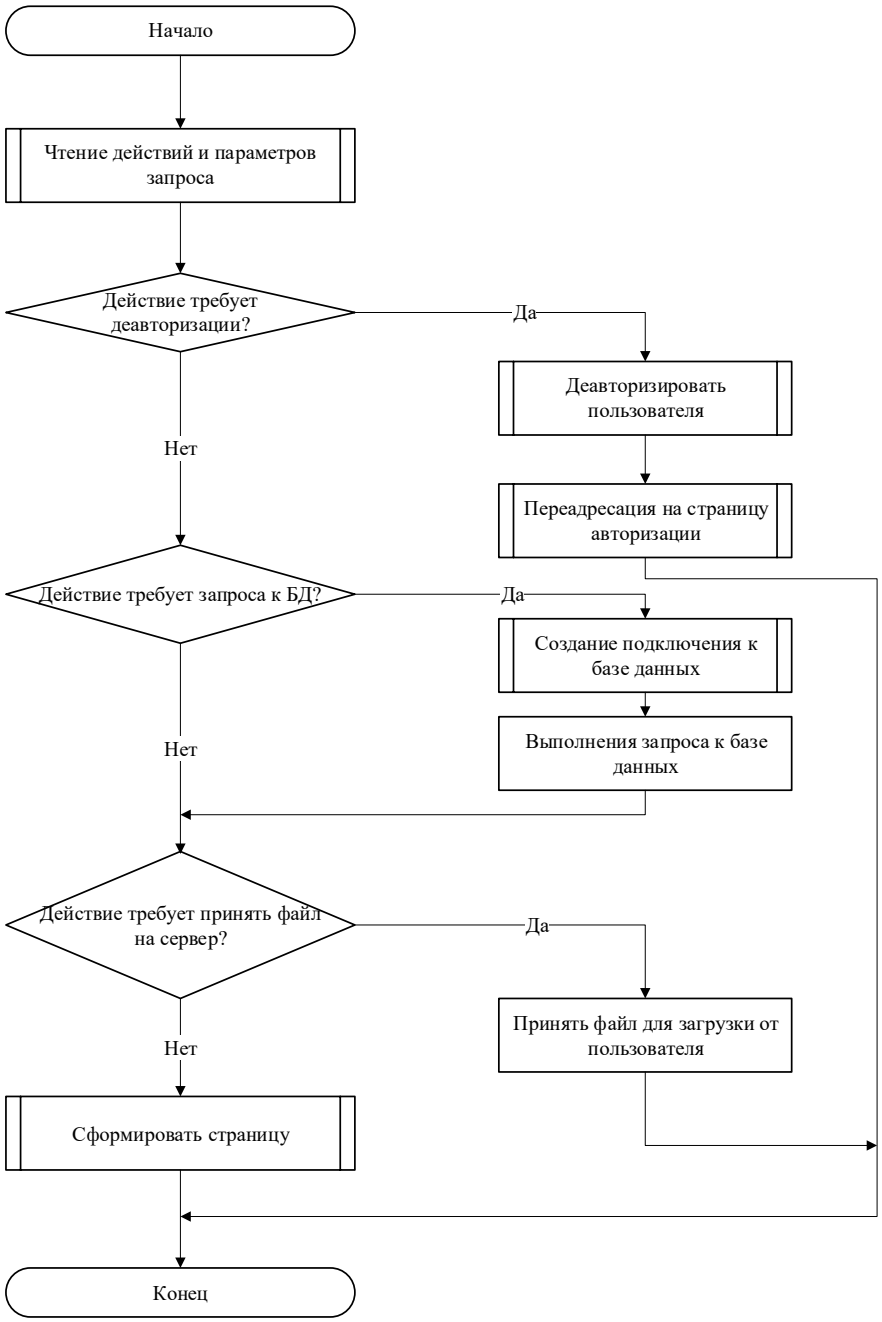


Рисунок 14 – Алгоритм вывода страницы

Если пользователь желает деавторизоваться, то выполняем процесс «деавторизации» и переадресуем его на страницу авторизации и заканчиваем работу алгоритма. Если же нет, то продолжаем работу. Далее проверяем, требуется ли сделать запрос к базе данных, если требуется, то создаем подключение к базе данных, после чего выполняем требуемый запрос. Дальше в не зависимости от того требуется сделать или нет, алгоритм продолжает работу.

4.2 Требования к программному продукту

Разрабатываемое программное обеспечение должно соответствовать следующим требованиям:

Требования к структуре и функционированию системы. В состав системы должны входить следующие подсистемы:

подсистема хранения данных;

подсистема управления тестами;

подсистема формирования и визуализации отчетности.

Подсистема хранения данных предназначена для хранения данных системы и данных для формирования отчетов.

Подсистема управления тестами предназначена для создания, редактирования, изменения теста, а также управления возможностью прохождения теста обучающимися.

Для организации информационного обмена между компонентами системы должны использоваться специальные протоколы прикладного уровня, такие как HTTP.

Требования к надежности. Система должна обеспечивать корректную обработку аварийных ситуаций, вызванных неверными действиями пользователей, неверным форматом или недопустимыми значениями входных данных. В указанных случаях пользователю должны выдаваться соответствующие аварийные сообщения, после чего возвращаться в рабочее состояние.

Требования к информационному обеспечению системы. Уровень хранения данных в системе должен быть построен на основе современных реляционных СУБД.

Для обеспечения целостности данных должны использоваться встроенные механизмы СУБД.

Доступ к данным должен предоставлен только авторизованным пользователям.

Структура БД должна быть организована рациональным способом.

Требования к лингвистическому обеспечению системы. При реализации системы должны применяться следующие языки высокого уровня: HTML, SQL и PHP.

Для организации диалога системы с пользователем должен применяться графический оконный пользовательский интерфейс.

Требования к программному обеспечению системы. Для функционирования приложения необходимо программное обеспечение, серверной части представленное в таблице 9.

Таблица 9 – Требования к конфигурации программного обеспечения серверной части

Операционная система	Microsoft Windows Server (2003/2008/2012)
СУБД	MySql 5.0
Web-сервер	Apache 2.0

Для функционирования приложения необходимо программное обеспечение, клиентской части представленное в таблице 10.

Таблица 10 – Требования к конфигурации программного обеспечения клиентской части

Операционная система	Microsoft Windows (7/8/8.1/10).
Общесистемное ПО	Internet Explorer, Google Chrome, Opera, Mozilla Firefox

Требования к техническому обеспечению системы. Для функционирования приложения необходимо аппаратное обеспечение, серверной части представленное в таблице 9.

Таблица 11 –Требования к конфигурации аппаратного обеспечения серверной части

Процессор	Intel Core 2 Duo 6450/2.3Ghz, Cache 4 Mb
Оперативная память (RAM)	4Гб SDRAM
Жесткий диск (доступного места на диске)	
Видеоадаптер	встроен в системную плату
CD–ROM	ATAPI CD–RW
Сетевая плата	Ethernet 100 Мб
Дополнительное оборудование	Монитор SVGA 1024x768, мышь, клавиатура

Для функционирования приложения необходимо аппаратное обеспечение, клиентской части представленное в таблице 12.

Таблица 12 – Требования к конфигурации аппаратного обеспечения клиентской части

Процессор	Intel Core 2 Duo 6450/2.3Ghz, Cache 4 Mb
Оперативная память	1Гб SDRAM
Видеоадаптер	встроен в системную плату
CD–ROM	ATAPI CD–RW
Сетевая плата	Ethernet 100 Мб
Дополнительное оборудование	Монитор SVGA 1024x768, мышь, клавиатура

4.3 Проектирование интерфейса программного продукта

Система тестирования собирает и хранит различную информацию о самом тестировании и результатах участников. В ней доступны как итоговые результаты участников, так и их детальные ответы на каждое из заданий, а также сводная информация по тестовым заданиям, позволяющая оценить их качество. Рассмотрим, как работать с результатами тестирования.

Вход в систему осуществляется по IP- адресу 10.0.0.10 в любом браузере Mozilla Firefox, Chrom, Opera.

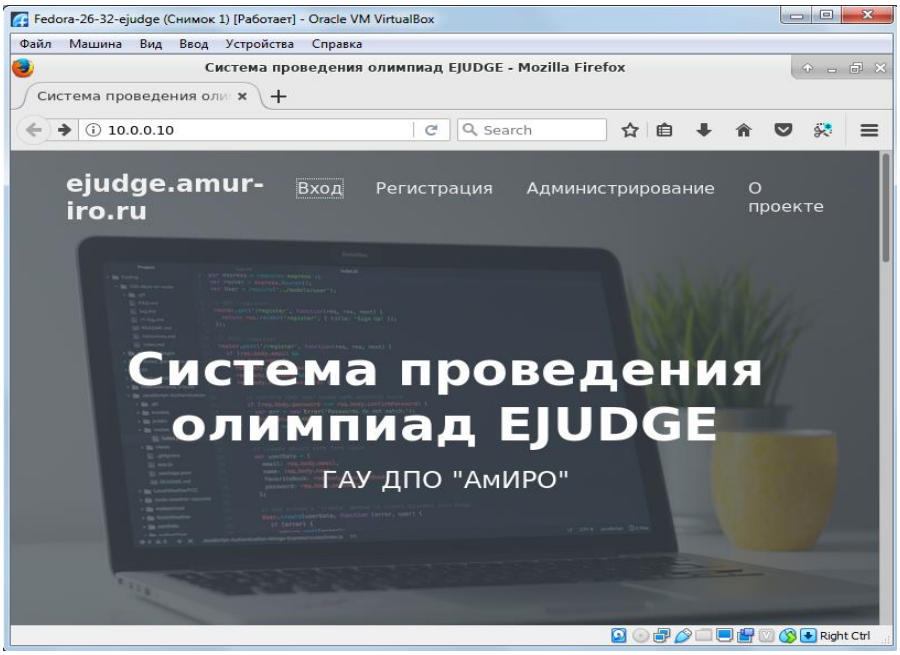


Рисунок 15 – Окно проведения олимпиады

На верхней панели 4 вкладки Вход, Регистрация, Администрирование, О проекте.

Вкладка вход для участника олимпиады, у которого уже есть пароль и регистрация участника находится в разработке.

В данной вкладке описание проекта об олимпиаде для чего нужна система автоматического тестирования, применение системы ejudge в учебном процессе, технические характеристики, текущее состояния проекта.

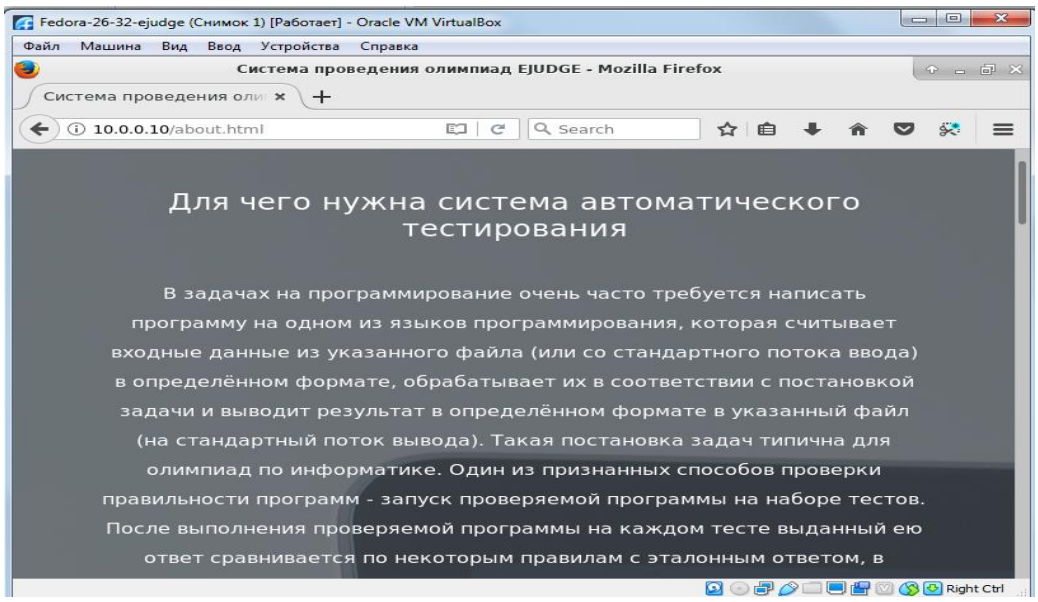


Рисунок 16 – О проекте

Вкладка администрирование вводим логин и пароль и входим в систему.

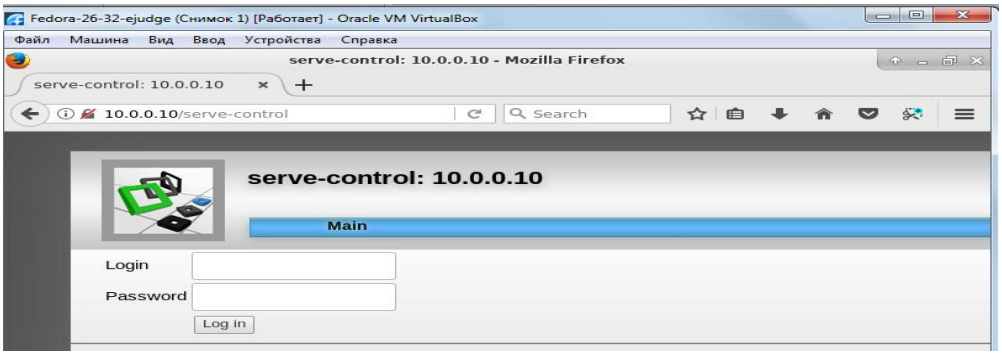


Рисунок 17 – Вход в администрирование

При входе в serve-control: ejudge@ 10.0.0.10 администратор добавляет Пробный тур, 1 тур, 2 тур для проведения олимпиады.

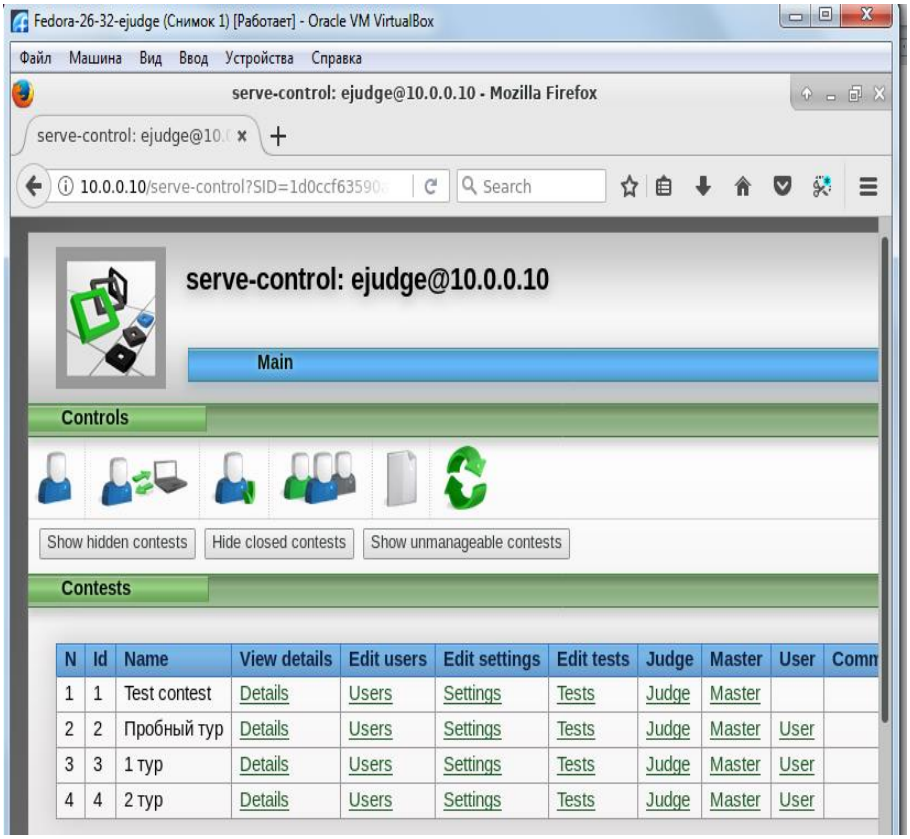


Рисунок 18 – serve-control: ejudge@ 10.0.0.10

Рассмотрим данные 2 тура User как решил задачи участник олимпиады. Четыре задачи удаление чисел, старая книга, красота фейерверка, обработка больших данных, за которые ученик мог получить 400 баллов. На данном рисунке видно сколько баллов набрал ученик за каждую задачу Score.

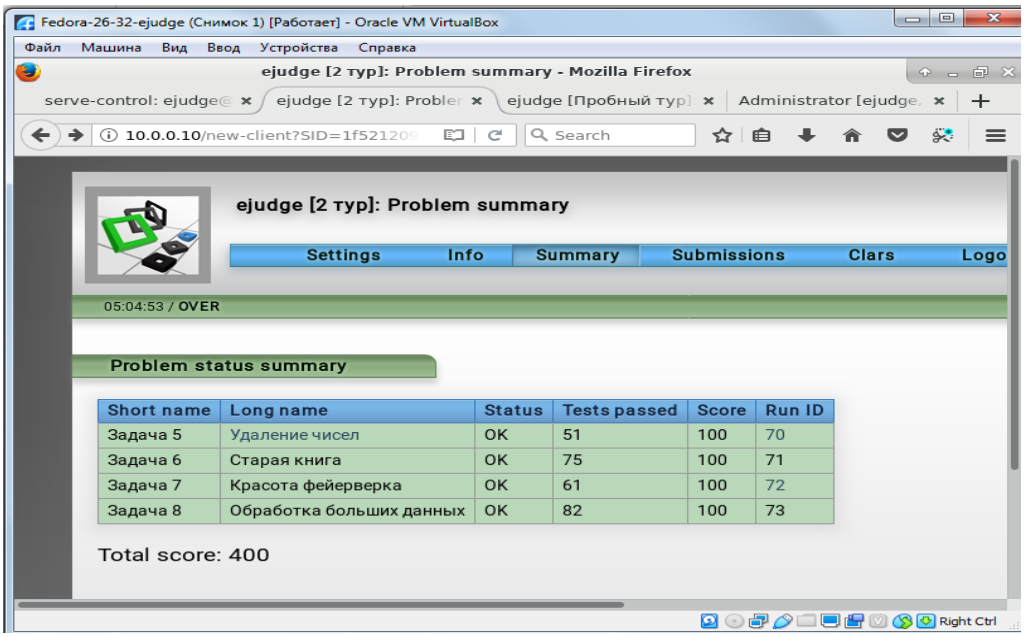


Рисунок 19 – Баллы за задачи

Рассмотрим на примере задачу 5 Удаление чисел Run ID 70. Чтобы проверить код задачи берется группа тестов с 2-51. На данном примере видим, что группа тестов использована по 5 за каждую группу тестов, выставлены баллы и участник набрал из 100 баллов 100

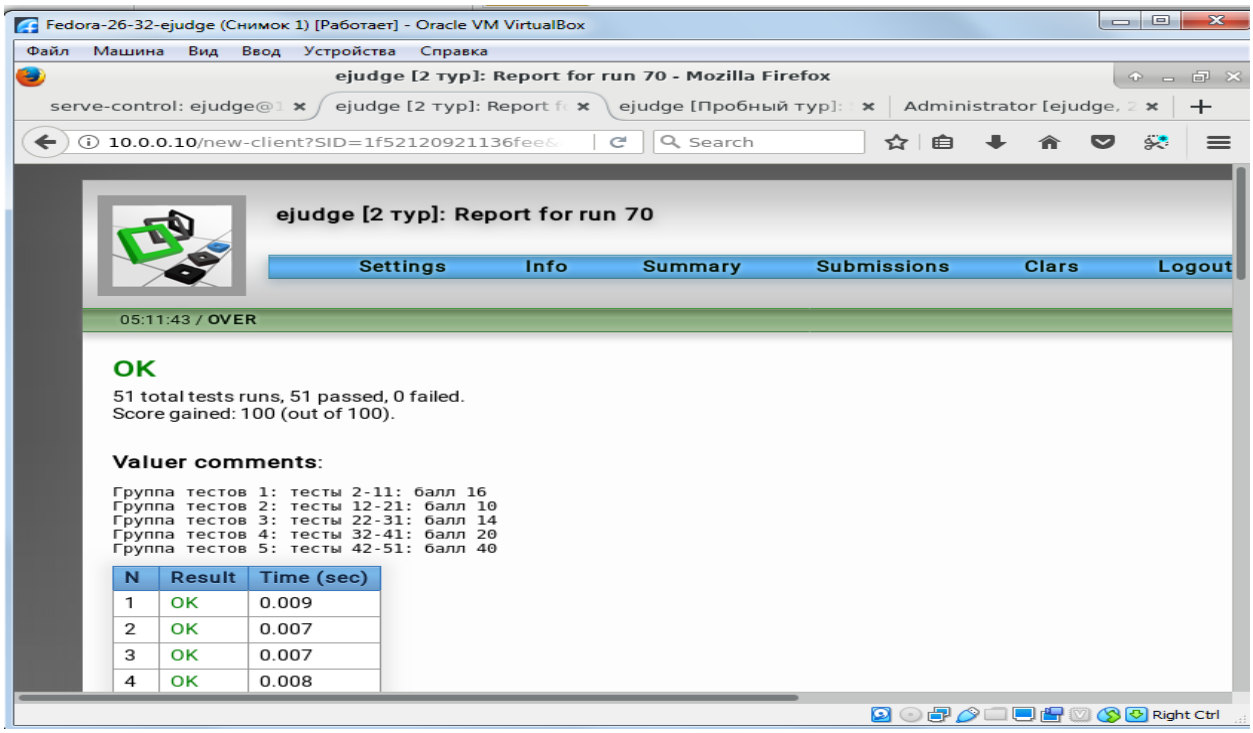
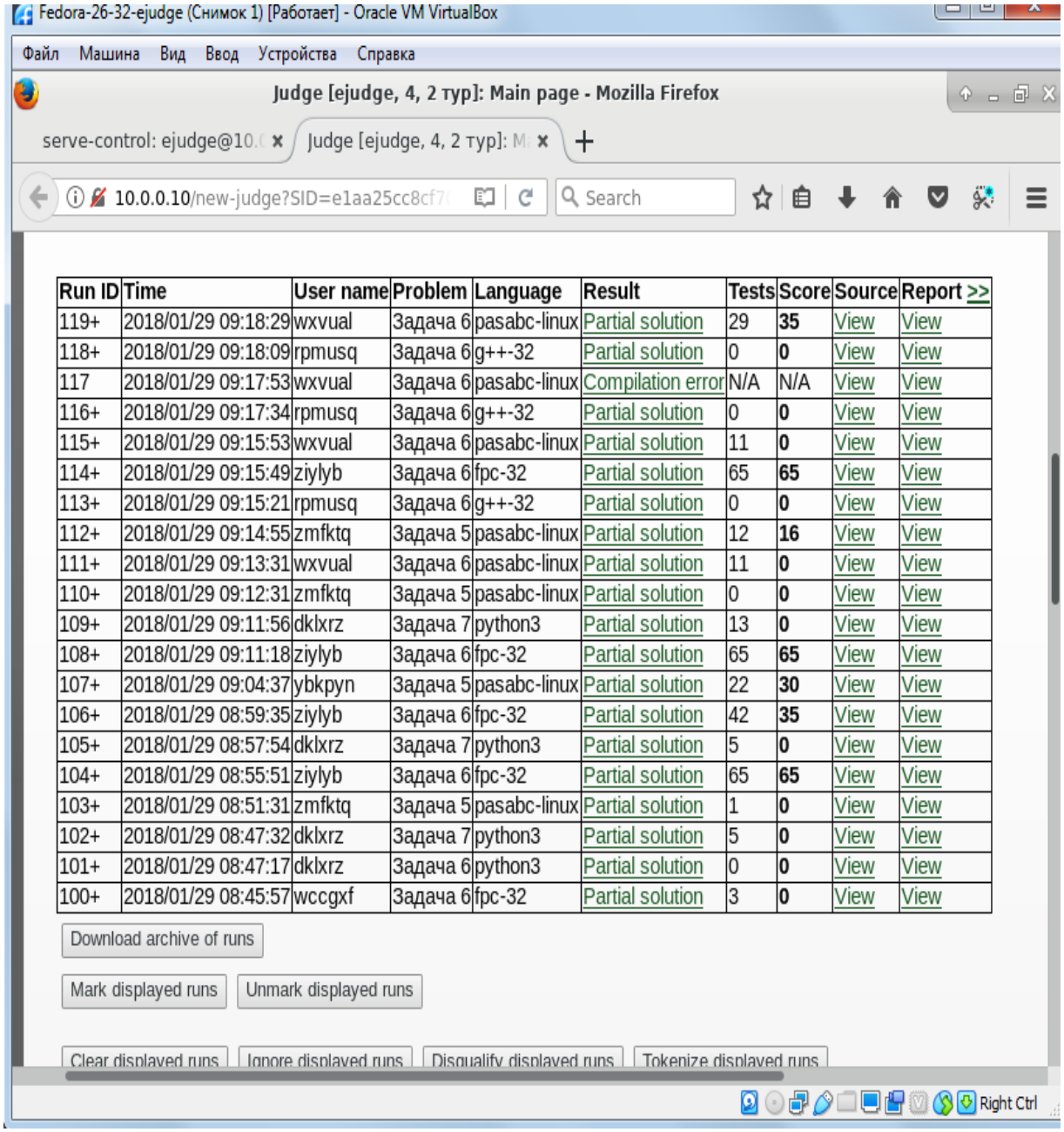


Рисунок 20 – Проверка задачи по тестам

Во вкладке Judge можно посмотреть весь список участников по User name кто какую задачу решает и сколько баллов набрал, а так же на каком языке программирования решена задача на данном рисунке представлены языки программирования Pascal ABC, СИ++, Free Pascal, Python.



Run ID	Time	User name	Problem	Language	Result	Tests	Score	Source	Report >>
119+	2018/01/29 09:18:29	wxvual	Задача 6	pasabc-linux	Partial solution	29	**35**	View	View
118+	2018/01/29 09:18:09	rpmusq	Задача 6	g++-32	Partial solution	0	**0**	View	View
117	2018/01/29 09:17:53	wxvual	Задача 6	pasabc-linux	Compilation error	N/A	N/A	View	View
116+	2018/01/29 09:17:34	rpmusq	Задача 6	g++-32	Partial solution	0	**0**	View	View
115+	2018/01/29 09:15:53	wxvual	Задача 6	pasabc-linux	Partial solution	11	**0**	View	View
114+	2018/01/29 09:15:49	ziylyb	Задача 6	fpc-32	Partial solution	65	**65**	View	View
113+	2018/01/29 09:15:21	rpmusq	Задача 6	g++-32	Partial solution	0	**0**	View	View
112+	2018/01/29 09:14:55	zmfktq	Задача 5	pasabc-linux	Partial solution	12	**16**	View	View
111+	2018/01/29 09:13:31	wxvual	Задача 6	pasabc-linux	Partial solution	11	**0**	View	View
110+	2018/01/29 09:12:31	zmfktq	Задача 5	pasabc-linux	Partial solution	0	**0**	View	View
109+	2018/01/29 09:11:56	dklxrz	Задача 7	python3	Partial solution	13	**0**	View	View
108+	2018/01/29 09:11:18	ziylyb	Задача 6	fpc-32	Partial solution	65	**65**	View	View
107+	2018/01/29 09:04:37	ybkpyn	Задача 5	pasabc-linux	Partial solution	22	**30**	View	View
106+	2018/01/29 08:59:35	ziylyb	Задача 6	fpc-32	Partial solution	42	**35**	View	View
105+	2018/01/29 08:57:54	dklxrz	Задача 7	python3	Partial solution	5	**0**	View	View
104+	2018/01/29 08:55:51	ziylyb	Задача 6	fpc-32	Partial solution	65	**65**	View	View
103+	2018/01/29 08:51:31	zmfktq	Задача 5	pasabc-linux	Partial solution	1	**0**	View	View
102+	2018/01/29 08:47:32	dklxrz	Задача 7	python3	Partial solution	5	**0**	View	View
101+	2018/01/29 08:47:17	dklxrz	Задача 6	python3	Partial solution	0	**0**	View	View
100+	2018/01/29 08:45:57	wccgxf	Задача 6	fpc-32	Partial solution	3	**0**	View	View

Рисунок 21 – Язык программирования задач

Во вкладке Source выбрав View можно просмотреть код программы, который прислал ученик на проверку по тестам.

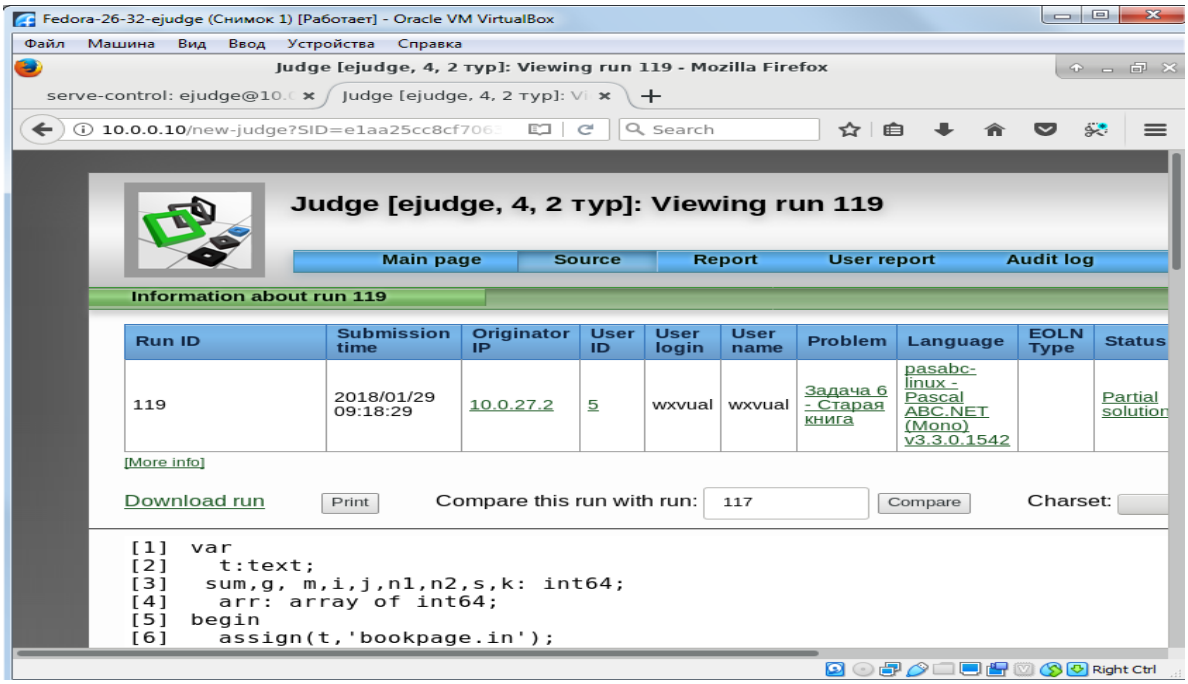


Рисунок 22 – Окно программного кода

Участник, набрав всего 35 баллов за задачу во вкладке Report подробное описание тестов.

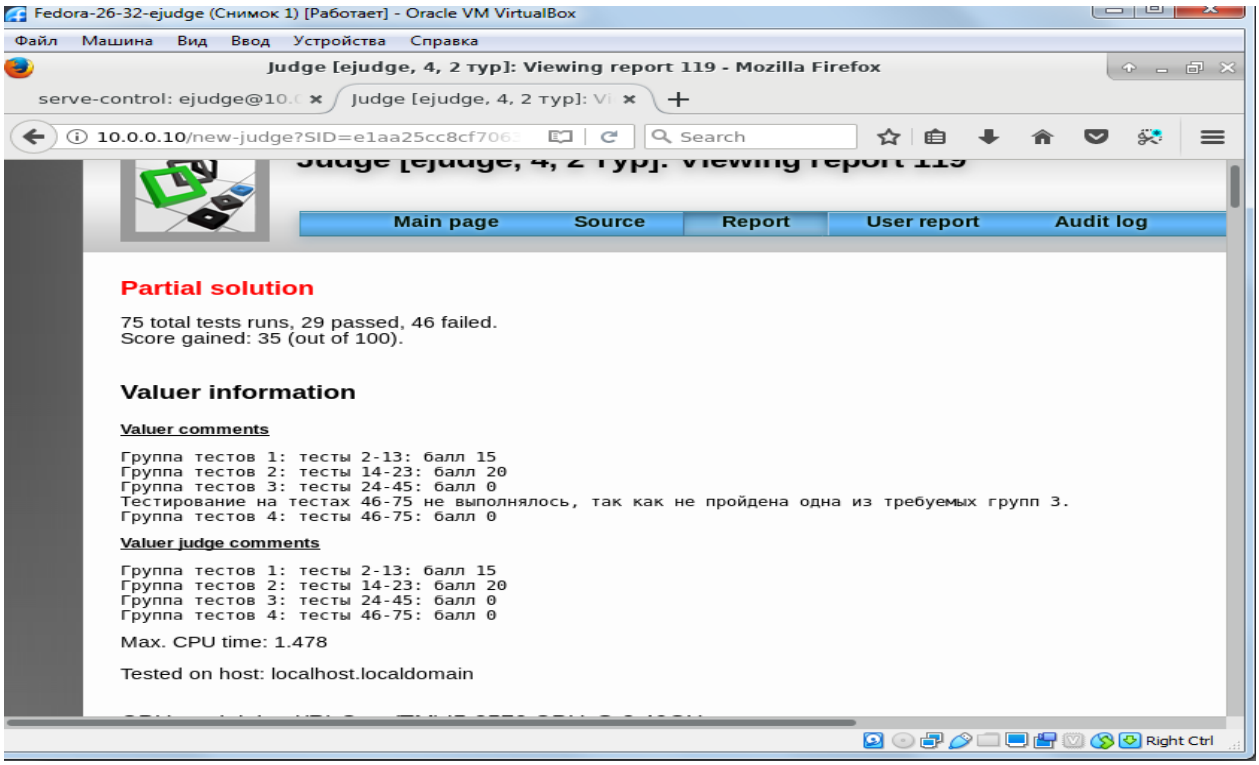
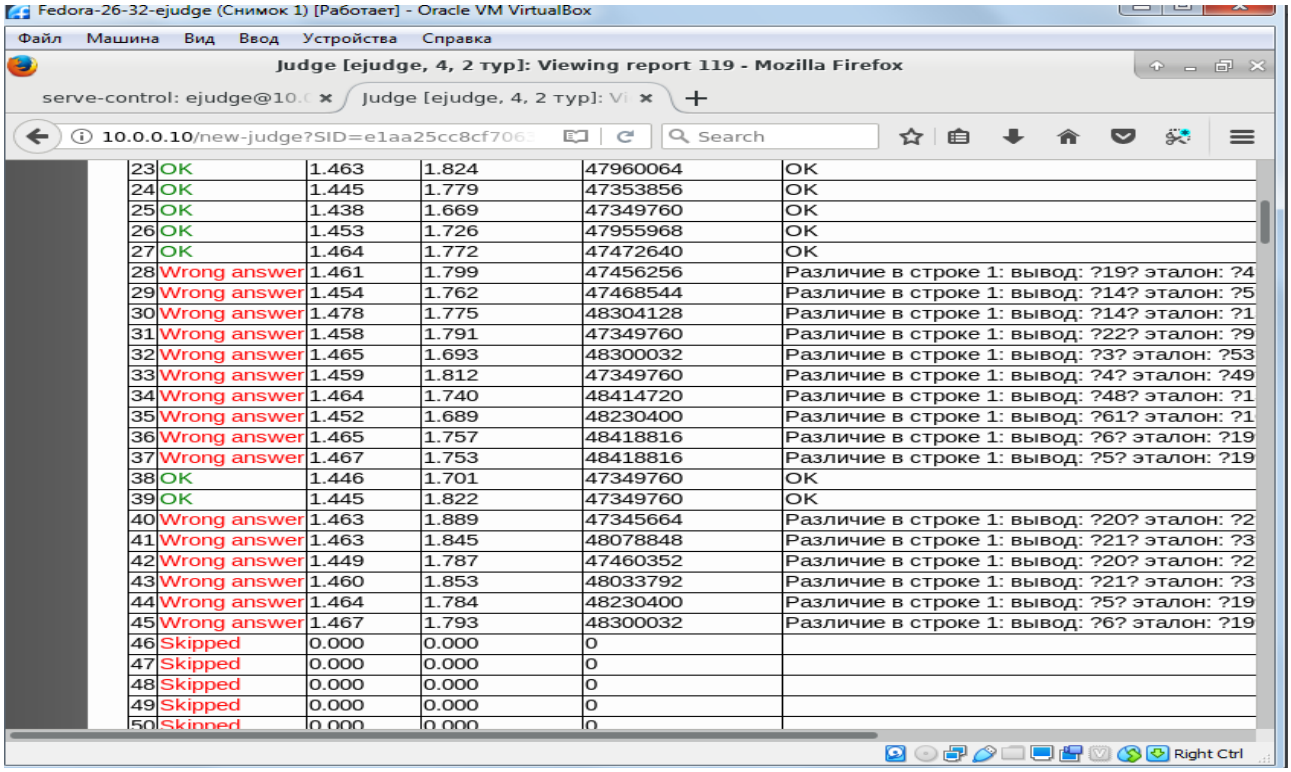


Рисунок 23 – Проверка по тестам



23	OK	1.463	1.824	47960064	OK
24	OK	1.445	1.779	47353856	OK
25	OK	1.438	1.669	47349760	OK
26	OK	1.453	1.726	47955968	OK
27	OK	1.464	1.772	47472640	OK
28	Wrong answer	1.461	1.799	47456256	Различие в строке 1: вывод: ?19? эталон: ?4
29	Wrong answer	1.454	1.762	47468544	Различие в строке 1: вывод: ?14? эталон: ?5
30	Wrong answer	1.478	1.775	48304128	Различие в строке 1: вывод: ?14? эталон: ?1
31	Wrong answer	1.458	1.791	47349760	Различие в строке 1: вывод: ?22? эталон: ?9
32	Wrong answer	1.465	1.693	48300032	Различие в строке 1: вывод: ?3? эталон: ?53
33	Wrong answer	1.459	1.812	47349760	Различие в строке 1: вывод: ?4? эталон: ?49
34	Wrong answer	1.464	1.740	48414720	Различие в строке 1: вывод: ?48? эталон: ?1
35	Wrong answer	1.452	1.689	48230400	Различие в строке 1: вывод: ?61? эталон: ?1
36	Wrong answer	1.465	1.757	48418816	Различие в строке 1: вывод: ?6? эталон: ?19
37	Wrong answer	1.467	1.753	48418816	Различие в строке 1: вывод: ?5? эталон: ?19
38	OK	1.446	1.701	47349760	OK
39	OK	1.445	1.822	47349760	OK
40	Wrong answer	1.463	1.889	47345664	Различие в строке 1: вывод: ?20? эталон: ?2
41	Wrong answer	1.463	1.845	48078848	Различие в строке 1: вывод: ?21? эталон: ?3
42	Wrong answer	1.449	1.787	47460352	Различие в строке 1: вывод: ?20? эталон: ?2
43	Wrong answer	1.460	1.853	48033792	Различие в строке 1: вывод: ?21? эталон: ?3
44	Wrong answer	1.464	1.784	48230400	Различие в строке 1: вывод: ?5? эталон: ?19
45	Wrong answer	1.467	1.793	48300032	Различие в строке 1: вывод: ?6? эталон: ?19
46	Skipped	0.000	0.000	0	
47	Skipped	0.000	0.000	0	
48	Skipped	0.000	0.000	0	
49	Skipped	0.000	0.000	0	
50	Skipped	0.000	0.000	0	

Рисунок 24 – Проверка по тестам ошибки

Тесты делаются заранее и загружаются в файловую систему программы, тесты имеют разное разрешение

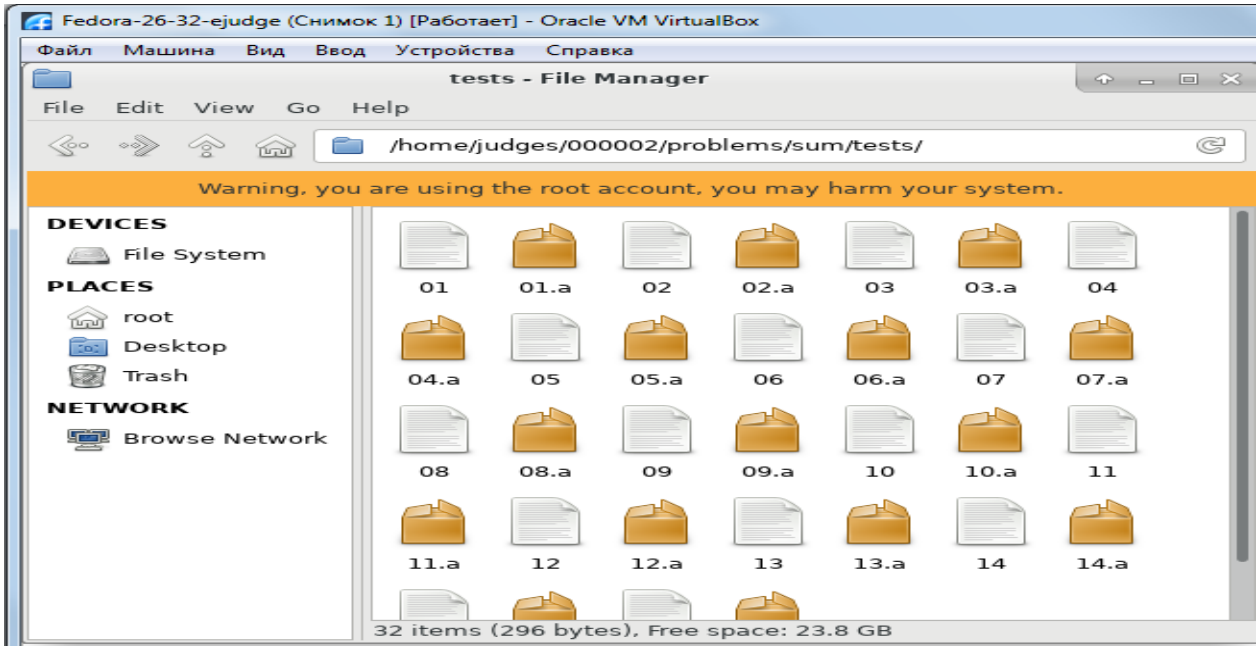


Рисунок 25 – Расположение тестов в программе

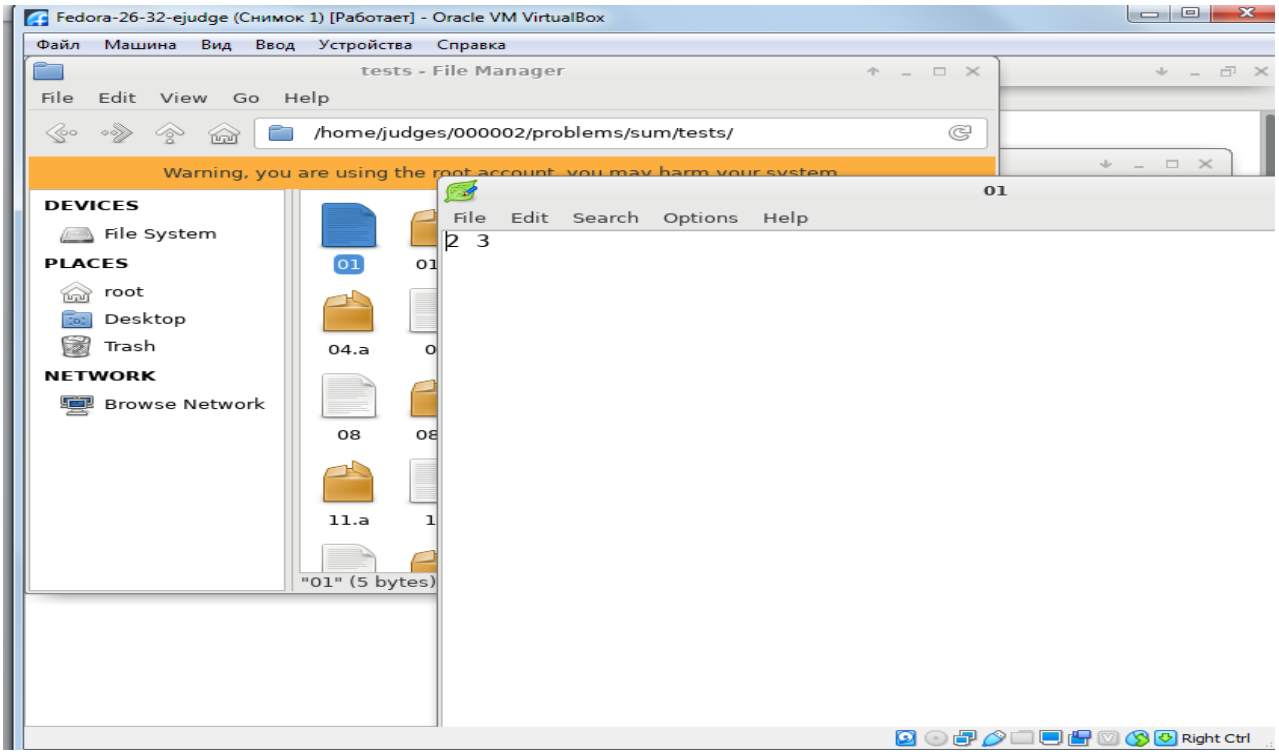


Рисунок 26 – Открытие теста 01

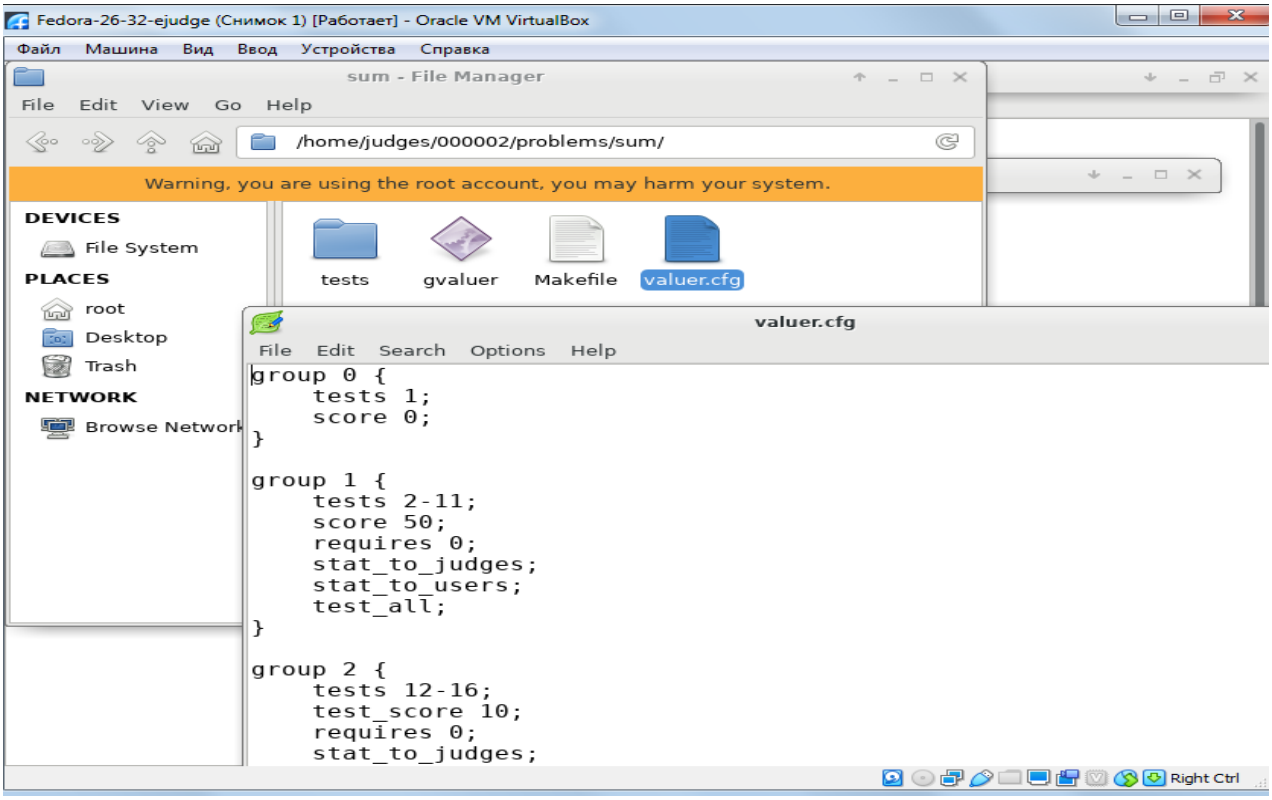


Рисунок 27 – Описание тестов по группам

Рассмотрели основные возможности использования результатов тестирования в системе ejudge, которая включает в себя весь спектр процессов тестирования.

4.4 Подробное описание разработанного программного продукта

Как выглядит олимпиадная задача по информатике

На данном этапе будет рассмотрена задача за 2018 год «Конец *K*-го урока».

Урок длится 45 минут, а перемена 5 или 10. В школе старт занятий происходит в 8 часов утра. Необходимо написать программу, которая выведет время, когда закончится k-ый урок.

В качестве входных данных передается одно целое неотрицательное число, которое меньше 15.

В результате вывод программы должен быть формата «hh mm», что показывает конец K-го урока.

Пример входных и выходных данных

Ввод	**Вывод**
1	8 45
6	12 55

Состав муниципальной задачи определяется ее условием. Сначала нужно написать условие задачи. Условие может включать в себя довольно длинный текст и быть несвязно с задачей. Главным и необходимым навыком при составлении олимпиадных задач является умение составлять модель задачи и придумывать последующие действия, которые приведут к ее решению.

В олимпиадных заданиях регионального уровня, задания составляются иначе. Сначала идут ограничения на задачу, например, это может быть время за которое необходимо, чтобы программа исполнилась. Как правило на одном тесте ограничение по времени составляет одну секунду. Так же есть ограничения на использование памяти. Оно составляет 64-256 мегабайт. Если для задачи

необходимо использовать файлы, то в начале указываются имена файлов, которые будут использоваться для хранения входных и выходных данных.

Файловый ввод-вывод не принимается на олимпиадах школьного уровня на всех этапах на олимпиаде по информатике.

Условие должно быть четко поставлено, не иметь неоднозначного понимания, а иметь ограниченное количество вариантов решения задачи. Ученик должен понимать что от него требуется и как это реализовать.

Вторая часть задачи – это изложение входных данных задачи. В данном примере на вход подается одно целое число. Существующие ограничения на входные данные указываются в описании формата.

В конкретном случае говорится, что число, которое подается на вход является целочисленным и неотрицательным, это число 15. Нужно дать понять ученикам, какие типы данных использовать при решении задачи.

Если входное число является слишком крупным, то для него понадобится long int а не integer. Эти моменты тоже необходимо учитывать при описании задачи.

Далее идет описание требований, что должна программа в итоге вывести. В текущем случае программа выводит количество часов и минут через пробел, когда заканчивается последний урок.

Программы на олимпиадах проверяются с помощью автоматической системы тестирования.

Для данной задачи единственным правильным решением будет 8 часов 45 минут для K=1.

В результате выполнения программы вывод должен быть таким: два целочисленных числ, которые разделены пробелом. Единственным правильным решением будет представление формата вывода, заданному формату, определенному в условии задачи.

В описании задачи приводятся примеры входных и выходных данных. Это облегчает восприятие требований к заданию. Например при вводе K, выводится ответ, который должен быть результирующим.

Проверка заданий осуществляется при помощи запуска тестов, которые подают на вход заранее подготовленные данные.

В результате программа должна считывать эти данные, решать задачу и выводить ответ. Происходит сравнение конечного результата, с результатом жюри. При совпадении ответа с верным ответом, задача считается успешно пройденной.

Для успешного решения задачи, программа должна пройти все тесты, которые для нее приготовлены. При прохождении всех тестов задача считается правильно решенной и ее автору присваивается наивысший балл. Если же какие то тесты не были успешно пройдены, то отведенное количество баллов для задачи делится на количество тестов и присваивается неполное количество баллов. Все действия по вычислению количества баллов и проверки происходят в автоматическом режиме, человек в них не участвует. Жюри только видит результаты проверки.

Как выглядит решение олимпиадной задачи

Решение задачи представляет собой программу, написанную на языке программирования, который поддерживает система тестирования. Обычно это известно еще до начала тестирования и на этапе подготовки участника к олимпиаде.

В тестирующей системе свой компилятор, который компилирует программный код, например на языке C++ или же Pascal, после чего происходит подача данных на вход и ожидание ответа от программы.

Если программа не использует файлы, то считывание данных происходит из стандартного потока ввода-вывода, функции, которые для этого используются – это стандартные функции. Например, для Python это ввода данных будет использована функция input, а для вывода print. Для Pascal read, write соответственно. Для языка Си это scanf, printf.

В системе тестирования происходит перенаправление стандартного ввода-вывода таким образом, данные на входе будут считываться из файлов, кото-

рые заранее были подготовлены. Таким образом получается, что система сначала считывает данные, а потом производит сравнение полученного результата, с заранее известными выходными данными.

В решениях не используется создание графических интерфейсов, взаимодействие программы с пользователем, в данном случае с тестирующей системой должно осуществляться с помощью консоли. Это обусловлено автоматической проверкой, которая невозможна для различных экранных форм и подобных интерфейсов, в том числе и веб-интерфейсов. Решение должно сопровождаться только взаимодействием с консолью.

Подробнее стоит сказать о веб-разработке. Данные не могут передаваться через Интернет, какими либо-запросами HTTP. Поэтому в задачах не используются языки программирования такие как PHP или Javascript, так как для их использования необходим веб-браузер.

Сначала происходит считывание входных данных. Программа может выводить некоторые необязательные сообщения.

Например, если выводится сообщение «Введите количество уроков», данный текст будет влиять на вывод всей программы,то есть будет проверен тестами, так как тестирующая система ожидает другого ввода, то это будет считаться ошибкой. Т.е в решении данной задачи не должно содержаться дополнительного вывода программы, которое будет некорректно распознано.

Так же не следует проверять входные данные на корректность, если этого не сказано в условии. В данном случае проверка числа 15 не несет в себе никакого смысла. Числа итак будут, как оговорено в условии меньше 15, значит проверка на то, что число меньше – бессмысленна.

Для успешной проверки тестирующей системой необходимо следовать строгому выводу, который задан в условии. Т.е в выводе не должны содержаться лишние символы, любые слова, например “Результат”. Тестирующая система в данном случае запрограммирована на получение двух чисел. И любой отличный ответ будет считаться неверным.

В некоторых случаях, чтобы после вывода ответа на дисплей программа сразу не закрывалась, выводят задержку при помощи pause или же вызывают функцию ввод. Этого делать не нужно, необходимо просто завершить программу после вывода. Ожидание каких-либо действий от пользователя – ошибочно.

Ниже приведены примеры, которые успешно, пройдут тесты в тестирующей системе на языке Паскаль, C, C++, C#:

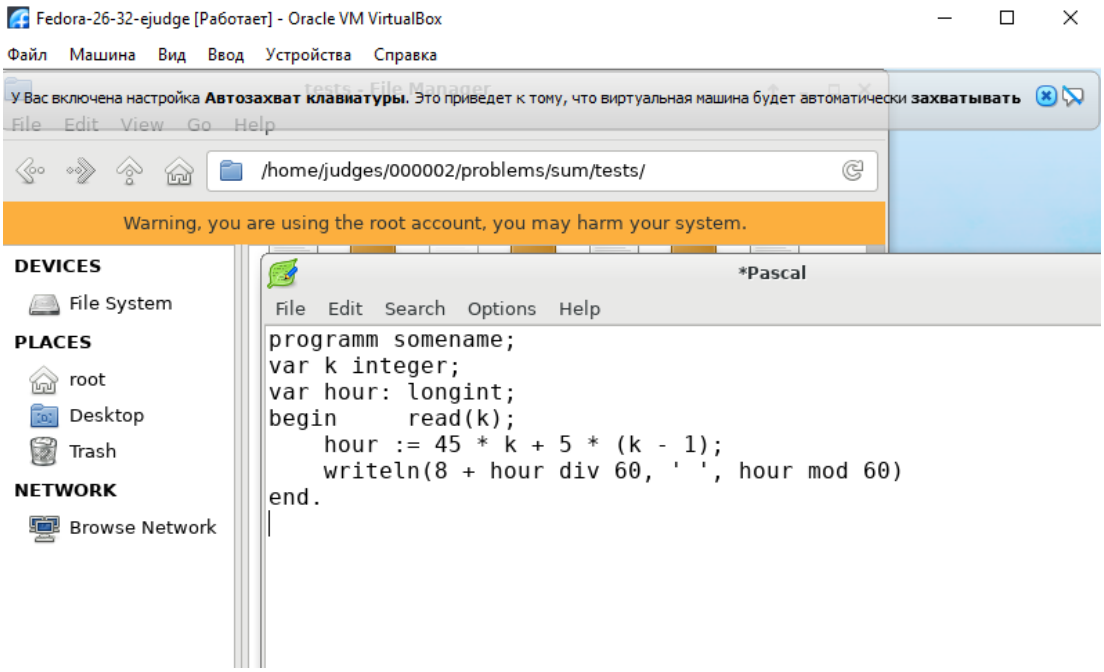
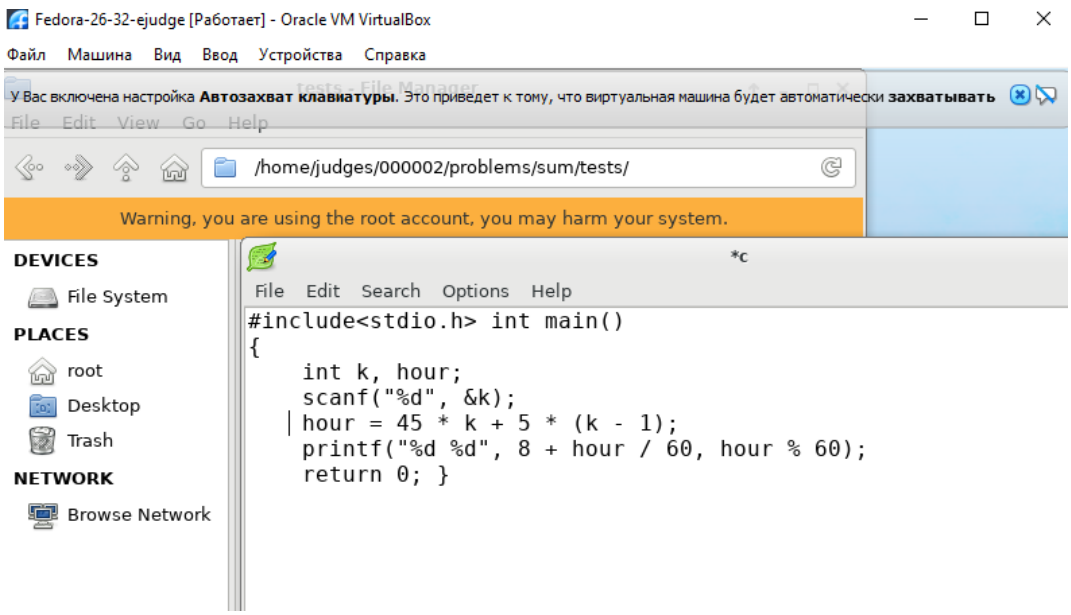


Рисунок 28 – Реализация программы на языке Паскаль



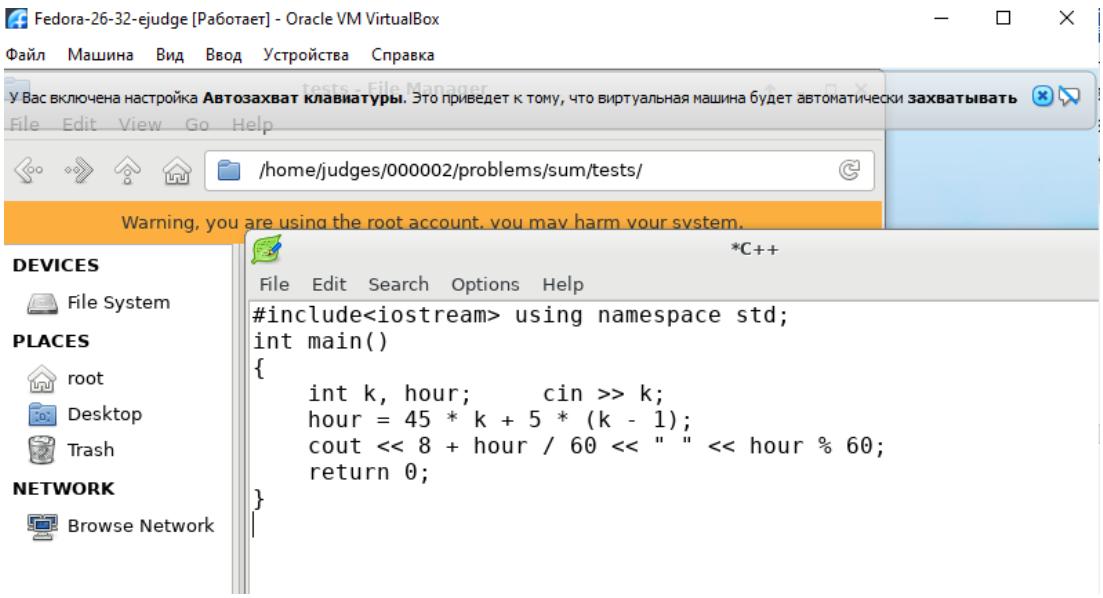


Рисунок 30 – Реализация программы на языке C++

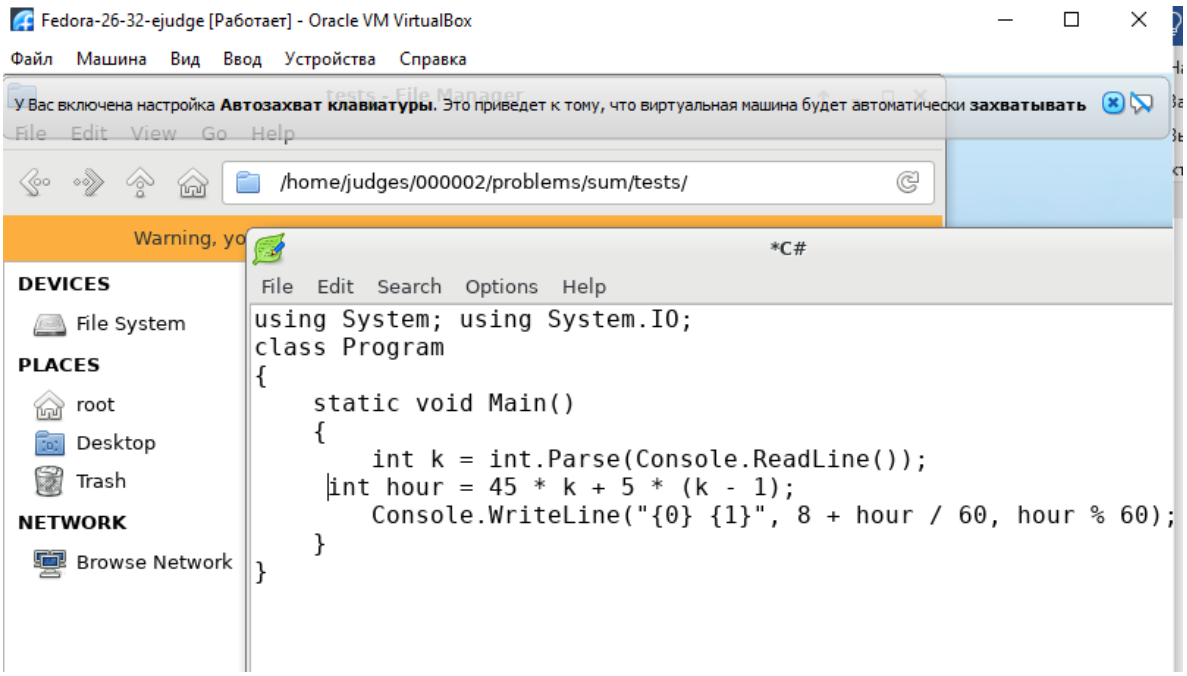


Рисунок 31 – Реализация программы на языке C#

Наиболее частые ошибки при использовании тестирующей системы.

В условии, которое прописано в задаче, приведен пример ввода-вывода. На этом примере основываются результаты тестов. В некоторых правилах олимпиады указывается, что решение будет проверено, если выводится правильный ответ на примерах из задания, данное условие есть на школьной и му-

ниципальной олимпиаде в Благовещенске.). При этом доступен полный протокол проверки на примерах тестов из условия, то есть в тестирующей системе будет показано, что вывела программа на тестах из условия.

После сдачи задания в тестирующую систему необходимо убедиться в том, что решение хотя бы проходит тесты из условия, а ещё лучше – если оно набирает много баллов). Если решение не проходит тесты из условия, то необходимо просмотреть протокол проверки, как правило, в протоколе содержится информация о причинах непрохождения программой тестов.

В большинстве случаев ошибки связанные с тем, что программа не может быть проверена тестирующей системой, имеют одну из следующих причин:

1. Программа не является консольным приложением, либо используются какие-то нестандартные возможности компилятора, привязанные к конкретной операционной системе, например, использование функции clrscr в Паскале, прекомилированных заголовочных файлов в Visual C++ и т. д.
2. В программах на Паскале используется модуль crt.
3. Программа выводит дополнительные сообщения, «улучшающие» интерфейс, например, «Введите количество уроков».
4. Программа выводит дополнительные сообщения, не предусмотренные форматом выходных данных, например, «часы», «минуты», «ответ».
5. Программа содержит «задержку» после окончания работы, то есть ждет от пользователя нажатия на какую-либо клавишу, ввода строки или иных действий.
6. Входные данные считываются не так как в условии задачи. Наиболее распространенная ошибка – в описании входных данных говорится, что входные числа находятся в разных строках, а пользователь вводит их в одной строке.
7. Программа должна вывести целое число, но из-за того, что в программе вместо целочисленного типа используется действительный тип данных например, тип real в языке Паскаль), программа выводит ответ в виде действительного числа, запись которого содержит точку или символ «e».

Ниже указаны особенности конкретных языков программирования и приведены примеры правильных программ на данных языках программирования.

Особенности использования разных языков программирования

Особенности использования языка Паскаль

Язык программирования Паскаль представлен в тестирующей системе компиляторами Free Pascal и PascalABC.Net.

В программах на языке Паскаль запрещается использовать модуль crt (даже подключение этого модуля может делает невозможным проверку программы).

Одной из типичных ошибок ввода-вывода при использовании языка Паскаль является неправильное использование функции ReadLn, которая после считывания данных считывает конец строки и пропускает все данные, находящиеся после считанных данных и до конца строки). Как правило это возникает в случае, когда программа получает на вход два или более чисел, записанных в одной строке через пробел. В этом случае нельзя читать данные при помощи ReadLn

a) ReadLn

b) так как такое использование функций ввода означает, что после первого числа должен быть конец строки, и следующее число записано в новой строке. Правильным чтением данных будет ReadLn

a, b) или Read a, Read b).

Если программа должна считывать только числа, то на языке Паскаль для чтения всегда лучше использовать функцию Read вместо ReadLn.

Особенности компилятора Free Pascal

Компилятор Free Pascal по умолчанию работает в 16-битном режиме, в частности, размер переменной integer составляет 2 байта 16 бит) и переменная такого типа может принимать значения от -32768 до 32767. Для хранения 32-битных целых чисел во Free Pascal следует использовать тип longint. Длина текстовой строки во Free Pascal ограничена 255 символами.

Данные параметры можно менять внутри программы при помощи директив компилятора, описанных в документации на язык Free Pascal.

В программах на Free Pascal нельзя использовать модуль crt и функции из этого модуля, например, ClrScr.

Особенности компилятора PascalABC.NET

В программах на PascalABC.NET нельзя использовать модуль crt и функции из этого модуля, например, ClrScr.

Особенности использования языков Си и C++

В тестирующей системе доступен компилятор GNU C/C++. Тестирующая система ejudge работает под управлением операционной системы Linux, поэтому компилятор Visual C/C++ тестирующей системой не поддерживается.

Для разработки программ можно использовать среду Visual C++, но проверять решения нужно под компилятором GNU C/C++. При создании проекта в Visual Studio меню File – New – Project) необходимо выбрать «Win32 Console Application». При создании проекта необходимо отключить использование прекомпилированных заголовочных файлов в окне диалога «Application Settings» убрать галочку «Additional options: Precompiled headers»). В готовом файле с программой не должно быть строки

#include "stdafx.h"

наличие этой строки означает, что не были отключены прекомпилированные заголовочные файлы). Также описание функции main необходимо изменить на int main()

вместо int _tmain(int argc, _TCHAR* argv[]) если среда Visual Studio создала такой шаблон для программы).

Программа на языках C или C++ должна заканчиваться с кодом возврата 0.

return 0, ненулевой код возврата может быть воспринят тестирующей системой, как ошибка в работе программы. Функция main должна возвращать значение типа int а не void).

Программа не должна содержать задержки после вывода результата типа вызова функции system (pause).

В программе необходимо явно подключать заголовочные файлы, т. к. компилятор в тестирующей системе не подключает автоматически ни одного заголовочного файла с функциями стандартной библиотеки. То есть в программах на языке С должно быть написано

#include <stdio.h>

В программах на языке С++ должно быть написано #include <iostream>

Особенности использования языка Python

На текущий момент существует несколько версий языка программирования Python. Эти версии существенно различаются. В основном это связано с кодировками и некоторыми функциями. Тестирующая система дает возможность выбрать версию – это или Python 2.7 или Python 3.6. Первая версия в новых задачах мало где используется. В основном используют python версии 3 и выше.

А среди минорных версий различий нет и ошибок при исполнении программ не должно возникнуть.

Если числа подаются на вход отдельно в каждой строке, то их необходимо считывать в отдельные переменные. А если все числа в одной строке, то можно считать в одну переменную, а затем методом split. Разделить сроку по пробелам и результат присвоить другим переменным.

Как выглядит тестирующая система.

Вход в тестирующую систему осуществляется путем авторизации по логину и паролю. Задания регистрируются в тестирующей системы и проверка осуществляется через нее же.

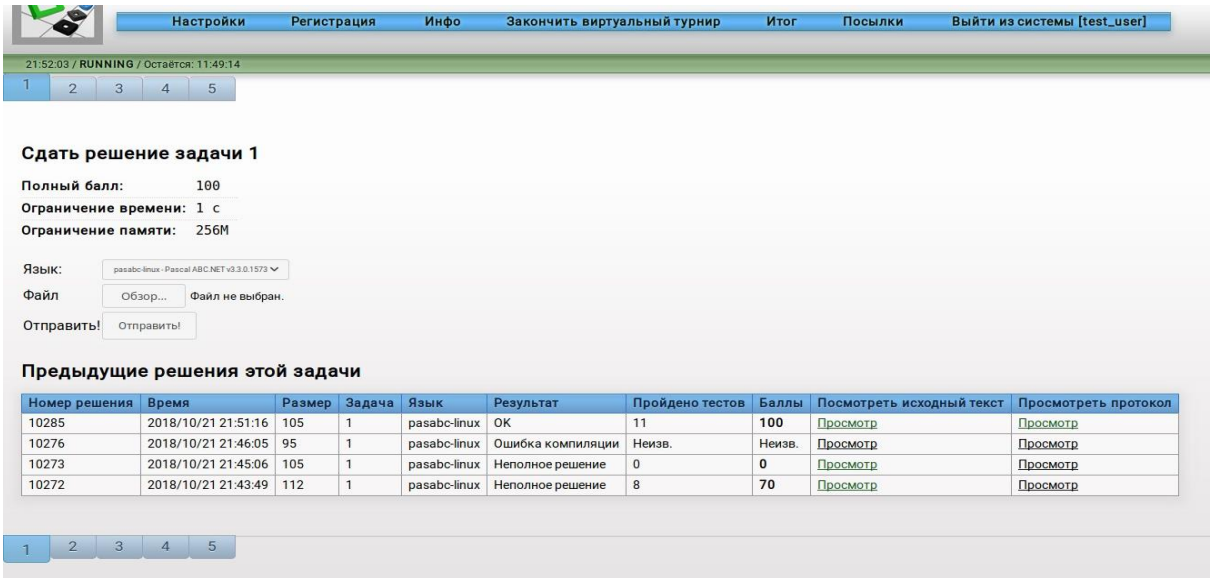


Рисунок 32 – Главная страница тестирующей системы во время олимпиады

На главной странице осуществляется сдача готовых решений по задачам а также их просмотр. Наверху экрана расположена навигация, которая состоит из ссылок меню.

– «Настройки» – в данном пункте меню можно выбрать язык интерфейса тестирующей системы.

– «Регистрация» – в этом пункте можно редактировать информацию, указанную при регстрации.

– «Инфо» – данный пункт содержит информациию о сервере и турнирах.

– «Закончить пробный турнир» – этот пункт нужен для окончания олимпиады. (для школьников 7-8 класса).

– «Итог» – отображает итоговую таблицу результатов.

– «Посылки» – отображает список задач, которые были отправлены на проверку.

Так же участник олимпиады может задать интересующий его вопрос непосредственно организационного характера или для уточнения условий задачи. Ход решения задачи сюда не относится. Вопросы необходимо задавать перед стартом. У него на странице имеется ссылка «Отправить вопрос». Далее в разделе сообщения, он может видеть ответ от организаторов.

По центру расположен блок с задачами. В шапке блока информация о статусе турнира. Исходное время, обратный отсчет времени.

Далее расположены вкладки, содержащие номера задач. Для прохождения той или иной задачи, нужно нажать на нужную вкладку.

Далее приведены ограничения задачи: это ограничение по расходуемой оперативной памяти, времени выполнения программы и стоимость задачи в баллах.

Ниже располагается форма для отправки задач на проверку.

Ниже приведена сетка решений по этой задачи другими участниками.

Сдача готового решения

Для сдачи работы необходимо загрузить файл с компьютера, содержащий решение. И отправить его.

Файл, который был выбран отправится на проверку в тестирующую систему, его решение должно появиться во вкладке «Предыдущее решение задачи»

Проверка работы осуществляется в течении минуты. Если была большая нагрузка на сервер, то может быть несколько миинут. Когда идет тестирование, задача находится в статусе «Обработка...» или «Выполнение...»

Поле с результатом будет содержать надпись OK, или ошибку или оповещение о неполном решении. Чтобы обновить страницу браузера необходимо нажать на кнопку f5. Или правой кнопкой мыши «Обновить».

В списке «Предыдущие решения этой задачи» содержатся:

– Идентификатор решения – уникальный номер решения, который был отправлен на проверку в текущем турнире.

– Фиксация времени отправки решения.

– Размер отправленного файла в байтах.

– Номер задачи, по которой сдано решение.

– Язык программирования, который был выбран в ходе решения задачи.

– Проверенное решение и итог по нему.

– Количество пройденных тестов данного решения.

– Количество баллов, которое было заработано в ходе решения задачи.

– Может не быть ссылки на исходное решение во время олимпиады.

– Ссылка для просмотра протокола решения данной задачи.

Вверху таблицы расположено решение, которое было сдано последним. На рисунке видно, что первая задача содержит 4 варианта решения. Наивысшее количество баллов получило последнее решение. Первое выполнено на 70 баллов, а второе и третье с ошибками.

Поле «Результат» может содержать несколько видов ответов.

Например «Ошибка компиляции» возникает, если в решении содержатся синтаксические ошибки и решение не скомпилировалось тестирующей системой. Причины, которые могли повлечь такой исход:

– Неверный синтаксис языка в программе, наличие ошибок.

– Указан ошибочный язык программирования.

– Файл ошибочный указан.

– Файл был отредактирован, но изменения не были приняты перед сдачей.

– Разные компиляторы например для языков Си. C++

Для подробного исследования проблемы необходимо просмотреть протокол решения. Он должен в себе содержать информацию, отображающуюся компилятором о наличии ошибок.

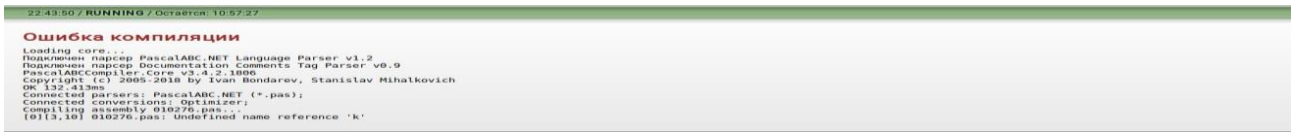


Рисунок 33 – Протокол проверки неполадок

На рисунке 33 имеется информация, что в написанной программе на языке PascalABC.NET в с третьей строке используется переменная «k», которая не была объявлена: [0][3,10] 010276.pas: Undefined name reference 'k'

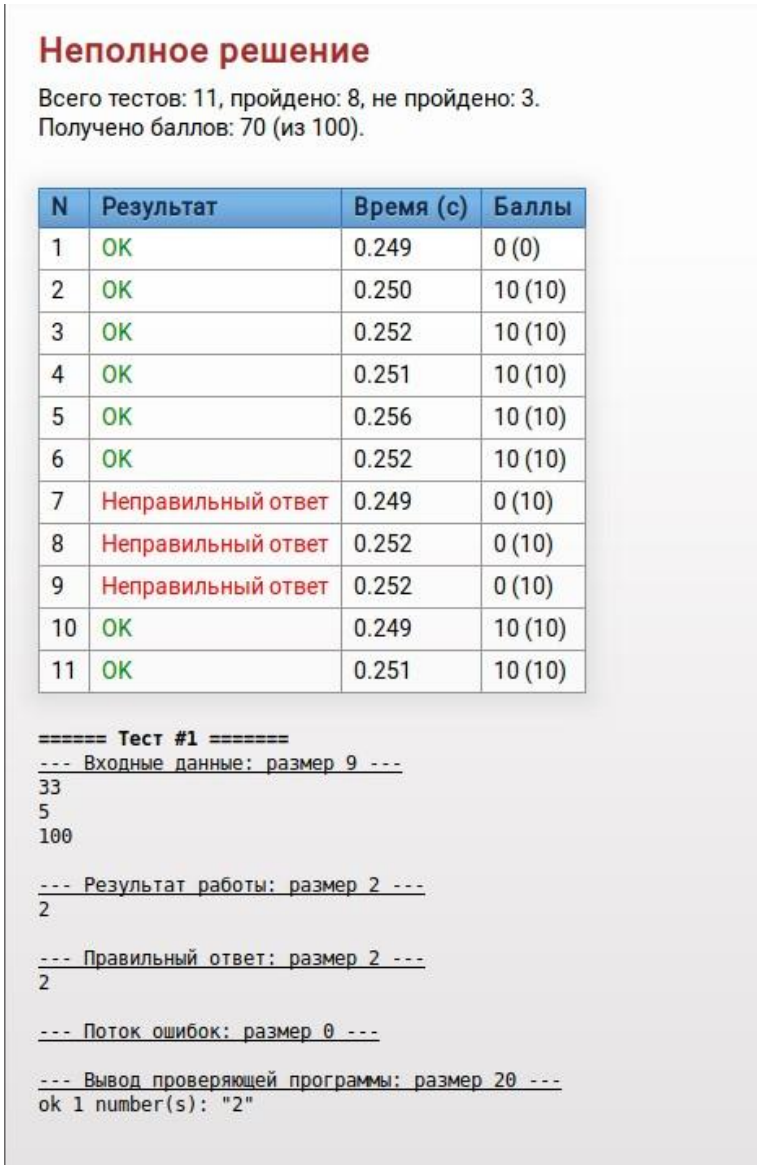
Решение не принято, необходимо пересмотреть решение и сдать задачу заново.

Также вывод может содержать только частичное решение, это означает, тесты не были пройдены в полной мере. И на каком то из тестов проихошел обрыв. Тогда «Результат» будет содержать «Частичное решение»

В сетки решений видны данные о количестве тестов, которые были успешно пройдены и баллы, которые за них получены. В то время как протокол проверки хранит в себе информацию более содержательную о решении.

Результат «OK» означает, что были пройдены все тесты и получен наивысший балл.

Просмотр протокола проверки



Неполное решение

Всего тестов: 11, пройдено: 8, не пройдено: 3.
Получено баллов: 70 (из 100).

N	Результат	Время (с)	Баллы
1	OK	0.249	0 (0)
2	OK	0.250	10 (10)
3	OK	0.252	10 (10)
4	OK	0.251	10 (10)
5	OK	0.256	10 (10)
6	OK	0.252	10 (10)
7	Неправильный ответ	0.249	0 (10)
8	Неправильный ответ	0.252	0 (10)
9	Неправильный ответ	0.252	0 (10)
10	OK	0.249	10 (10)
11	OK	0.251	10 (10)

```
====== Тест #1 =======
--- Входные данные: размер 9 ---
33
5
100

--- Результат работы: размер 2 ---
2

--- Правильный ответ: размер 2 ---
2

--- Поток ошибок: размер 0 ---

--- Вывод проверяющей программы: размер 20 ---
ok 1 number(s): "2"
```

Рисунок 34 – Протокол проверки

Протокол проверки предоставляет следующую информацию.

В таблице с верху указан номер теста в которой идет для каждого теста, результат запуска программы на этом тесте OK», «Неправильный ответ», «Превышено максимальное время работы» и т. д.), время работы программы на этом тесте в секундах, количество баллов, которое получила программа за прохождение этого теста в скобках – максимальное количество баллов за этот тест). На изображении ниже приведён пример решения, которое было проверено на 11 тестах. Тест № 1 – пример из условия задачи, он оценивается в 0 баллов. Следующие 10 тестов оцениваются в 10 баллов каждый. Программа прошла все тесты, кроме трёх №№ 7, 8, 9), поэтому программа оценивается в 70 баллов.

Поскольку тест № 1 является примером из условия задачи, то ниже приведён полный протокол проверки на тесте № 1. Это протокол содержит разделы:

– «Входные данные» – то, что было подано на вход программе.

– «Результат работы» – то, что вывела программа на этих входных данных при запуске в тестирующей системе.

– «Правильный ответ» – что является правильным ответом для данного набора входных данных.

– «Поток ошибок» – что программа вывела в стандартный поток сообщений об ошибках stderr, например, в программах на языке C++ в этот поток вывод осуществляется при помощи cerr << ...), что может использоваться для вывода отладочной информации. Вывод программы тестирующая система игнорирует в стандартный поток сообщений об ошибках.

– «Проверяющая программа «Ввывод»» – здесь может содержаться полезная информация, которую вывела программа, используемая для проверки ответа на правильность.

В данном примере проверяющая программа вывела сообщение о том, что в ответе указано 1 число, это число «2», и это правильный ответ.

Приведём пример протокола, в котором решение не проходит пример из условия.

Неполное решение

Всего тестов: 11, пройдено: 0, не пройдено: 11.
Получено баллов: 0 (из 100).

Комментарии оценивания:

Тестирование на тестах 2-11 не выполнялось, так как не пройдена одна из требуемых групп 0.

N	Результат	Время (с)
1	Неправильный формат вывода	0.257
2	Пропущен	0.000
3	Пропущен	0.000
4	Пропущен	0.000
5	Пропущен	0.000
6	Пропущен	0.000
7	Пропущен	0.000
8	Пропущен	0.000
9	Пропущен	0.000
10	Пропущен	0.000
11	Пропущен	0.000

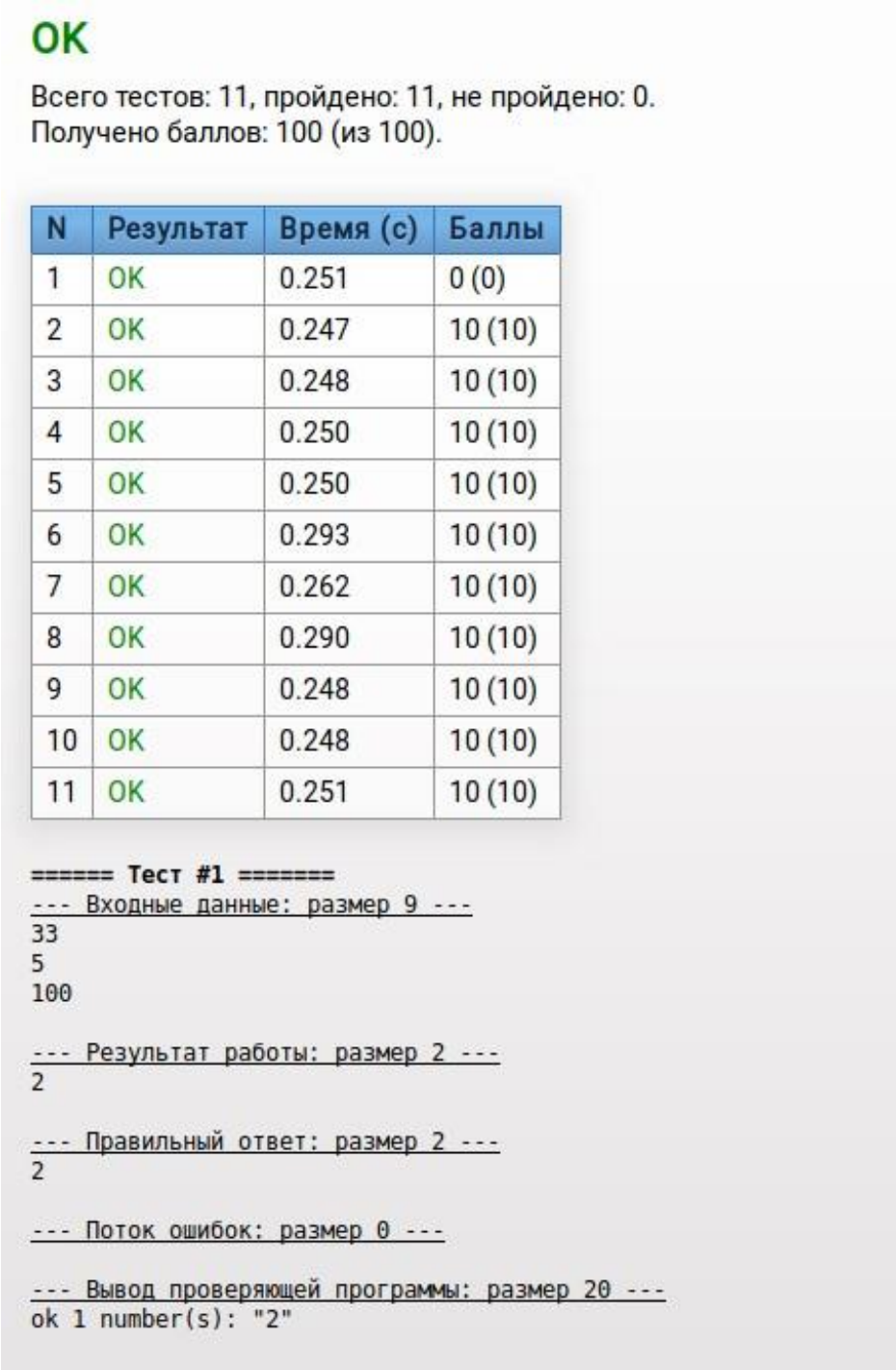
```
====== Тест #1 =======
--- Входные данные: размер 9 ---
33
5
100

--- Результат работы: размер 17 ---
2.88600288600289

--- Правильный ответ: размер 2 ---
2

--- Поток ошибок: размер 0 ---

--- Вывод проверяющей программы: размер 67 ---
wrong output format Expected integer, but "2.88600288600289" found
```

Рисунок 35 – Протокол проверки неполное решение

В этом примере программа вывела в качестве ответа действительное число 2.88600288600289, а по условию задачи программа должна вывести целое число, поэтому результат проверки – «неправильный формат вывода». Отметим, что если в условии задачи содержится требование о том, что программа должна вывести целое число, то вывод «2.0», то есть вывод действительного числа, равного целому числу 2, также будет считаться неправильным. Для те-

стов 2-11 указан результат «Пропущен». Это означает, что тестирование на этих тестах не выполнялось, потому что программа не прошла тест из условия задачи.

Приведём пример протокола проверки решения, проходящего все тесты.



OK

Всего тестов: 11, пройдено: 11, не пройдено: 0.
Получено баллов: 100 (из 100).

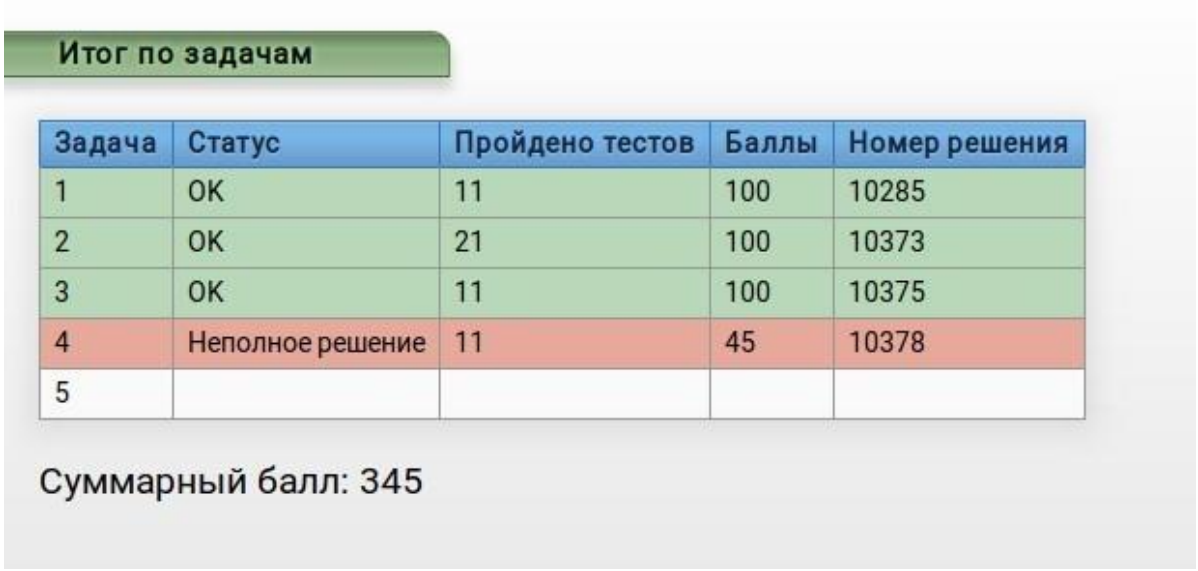
N	Результат	Время (с)	Баллы
1	OK	0.251	0 (0)
2	OK	0.247	10 (10)
3	OK	0.248	10 (10)
4	OK	0.250	10 (10)
5	OK	0.250	10 (10)
6	OK	0.293	10 (10)
7	OK	0.262	10 (10)
8	OK	0.290	10 (10)
9	OK	0.248	10 (10)
10	OK	0.248	10 (10)
11	OK	0.251	10 (10)

```
====== Тест #1 =======
--- Входные данные: размер 9 ---
33
5
100

--- Результат работы: размер 2 ---
2

--- Правильный ответ: размер 2 ---
2

--- Поток ошибок: размер 0 ---

--- Вывод проверяющей программы: размер 20 ---
ok 1 number(s): "2"
```

Рисунок 36 – Протокол проверки проходящего все тесты

Итог просмотра по всем задачам

Таблица с результатом сдачи решений по всем задачам отображается во вкладке «Итог».



Итог по задачам

Задача	Статус	Пройдено тестов	Баллы	Номер решения
1	OK	11	100	10285
2	OK	21	100	10373
3	OK	11	100	10375
4	Неполное решение	11	45	10378
5				

Суммарный балл: 345

Рисунок 37 – Итог по задачам

В этом примере по первым трём задачам были сданы решения на 100 баллов, по четвёртой задаче – решение на 45 баллов, по пятой задаче ничего не было сдано. По каждой задаче засчитывается наилучшее решение.

Список можно посмотреть всех сданных решений по всем задачам во вкладке «Посылки.

Отправленные решения (последние 15)

Номер решения	Время	Размер	Задача	Язык	Результат	Пройдено тестов	Баллы	Посмотреть исходный текст	Просмотреть протокол
10378	2018/10/21 23:19:56	364	4	python3	Неполное решение	11	**45**	Просмотр	Просмотр
10375	2018/10/21 23:19:31	165	3	python3	OK	11	**100**	Просмотр	Просмотр
10373	2018/10/21 23:19:11	370	2	python3	OK	21	**100**	Просмотр	Просмотр
10328	2018/10/21 22:20:03	107	1	pasabc-linux	OK	11	**100**	Просмотр	Просмотр
10325	2018/10/21 22:19:18	106	1	pasabc-linux	OK	11	**100**	Просмотр	Просмотр
10285	2018/10/21 21:51:16	105	1	pasabc-linux	OK	11	**100**	Просмотр	Просмотр
10276	2018/10/21 21:46:05	95	1	pasabc-linux	Ошибка компиляции	Неизв.	Неизв.	Просмотр	Просмотр
10273	2018/10/21 21:45:06	105	1	pasabc-linux	Неполное решение	0	**0**	Просмотр	Просмотр
10272	2018/10/21 21:43:49	112	1	pasabc-linux	Неполное решение	8	**70**	Просмотр	Просмотр

Посмотреть всё

Рисунок 38 – Отправленные решения

Окончание олимпиады

Когда заканчивается олимпиада, то обычно наиболее общие протоколы проверки становятся доступными. В них приведены данные входа и выхода, так же верные ответы для всех тестов.

Допустимые результаты проверки на одном тесте

Перечисление заключений, которые возможны для отдельно взятого теста из тестирующей системы.

OK

Тест пройден корректно, о чем и сообщила программа на этом тесте.

Неверный ответ

В тесте на данном этапе прохождения содержатся ошибки ответа, о чем и сообщила программа на текущем тесте.

Неверный формат вывода

Результат вывода программы разнится с необходимым описанием формата данных на выходе, которые приведены в условии задачи. Причины, которые привели к данному результату:

1. Вывод программы пустой, это могло произойти по причине наличия ошибок в программе или имени файла на выходе при использовании системы ввода-вывода.
2. Выводятся ненужные сообщения, например, «Введите число» или «Ответ».
3. Несоответствие содержания вывода и типа данных на выходе, например программа должна вывести несколько чисел, а выводит текст.
4. Наличие в выводе лишних пробельных символов и разделителей строк.
5. Проблема при привидении типа из числа с плавающей точкой к целочисленному числу. Например, ожидается вывод дробного числа с двумя знаками после запятой, а выводится целочисленное.

Ошибка исполнения

Выброс исключения при выполнении некорректной операции, в ходе тестирования программы.

1. Недопустимое математическая операция, к таким относятся: деление числа на ноль, взятие корня из отрицательного числа, переполнение переменных.

2. Часты ошибки при работе с памятью – это обращение к несуществующему элементу массива, обращение к несуществующим областям памяти, т.е. Некорректная работа с указателями. Отсутствие уничтожение объекта, освобождения памяти после выделения, например не закрыли файл после чтения или записи.

3. Чтение некорректных входных данных приводит к ошибке – например, на вход программы подаются числа, которые записаны каждое на новой строке, а при чтении ожидается, что разделителем чисел является пробел, а не перенос строки.

4. Любые синтаксические ошибки, еще до исполнения самой программы отлавливаются. Например, это обращение к несуществующей переменной или к переменной, которая не была ранее объявлена.

5. Рекурсивные функции.

6. Если явно указать код завершения программы отличный от 0 в языках программирования Си или C++.

Превышение максимального времени работы программы

Отсутствие исполнения программы в заданное время. Причинами такого поведения являются:

1. Нарушен алгоритм программы – выполнение бесконечного цикла.

2. Отсутствие оптимизации программы – время выполнения недопустимо длинное.

3. Возникновение ошибок при чтении с клавиатуры, например, программа может ожидать на входе два числа, но введено будет только одно число, тем самым программа будет долго ожидать ввода второго числа.

4. Так же программа может ожидать нажатия на клавишу, после, например, завершения теста. .

Превышение лимита памяти

Если в задаче есть ограничения на используемую оперативную память, это приведет к ошибке. Причины, которые могут это вызвать:

1. Если используются очень большие массивы данных или иные структуры.

2. Рекурсивные функции.

3. Некорректное обращение к указателя в C или C++ может вызывать исключение, как «Превышение лимита памяти».

Пропущен

Отсутствие проверки на данном тесте. Если предыдущие тесты не были пройдены, то проверка на последующих не производится.

Задания с вводом ответа для 7-8 классов

На школьном и муниципальном этапе олимпиады для 7-8 классов в г. Москве предлагаются задания с вводом ответа. Для выполнения этих заданий не нужно уметь программировать, ответом на задание является текст, который вводится в однострочное или многострочное поле ввода. Рассмотрим пример такой задачи. Дан список чисел:

3, 7, 1, 6, 2, 4, 8, 5.

Разрешается за одну операцию поменять местами два любых числа в этом списке. Например, если поменять местами числа 6 и 8, то получится список 3, 7, 1, 8, 2, 4, 6, 5.

Упорядочите этот список по возрастанию, то есть получите из него список 1, 2, 3, 4, 5, 6, 7, 8, используя минимальное число обменов.

Решение этой задачи нужно записать в виде последовательности обменов, каждый обмен записывается в одной строке. Один обмен записывается в виде двух различных чисел от 1 до 8, которые нужно поменять местами, записанных через пробел пример: 6 8).

Для сдачи этой задачи необходимо ввести ответ в поле ввода, затем нажать на кнопку «Отправить!».

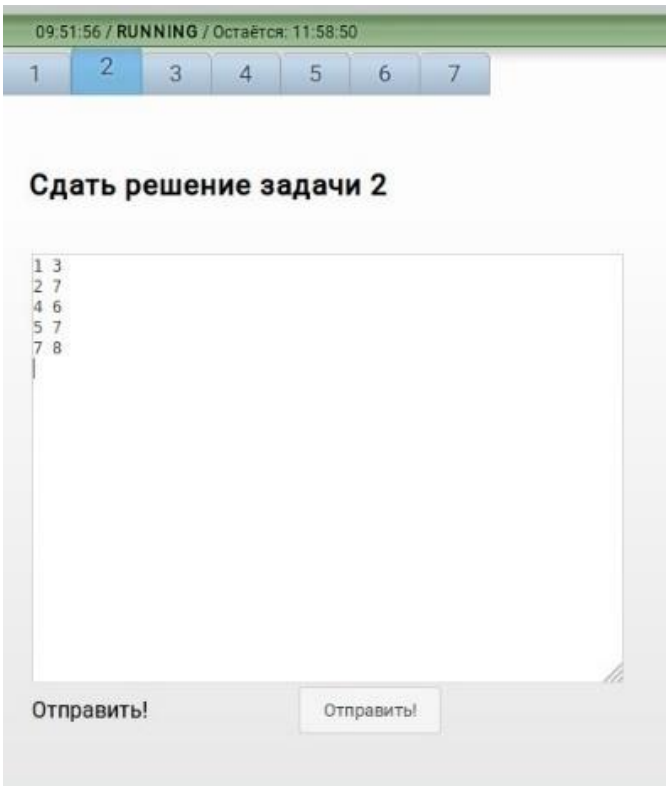


Рисунок 39 – Окно сдачи задачи

Условие задачи предполагает много возможных правильных ответов нет единственно возможного правильного ответа), поэтому для проверки ответа также используется проверяющая программа. Для того, чтобы решение могло быть проверено, необходимо, чтобы оно соответствовала требованиям к формату записи ответа, описанному в условии задачи. В условии этой задачи написано, что в ответе должно быть несколько строк, в каждой строке записано два различных числа от 1 до 8 через пробел. Никаких других символов в строке быть не должно.

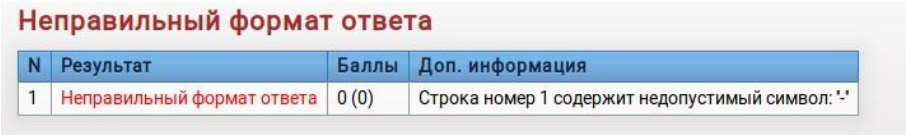
После отправки ответа, осуществляется проверка на соответствие сданного ответа формату, описанному в условии задачи. Если форма записи ответа не соответствует условию задачи, то результатом проверки будет «Неправильный формат ответа». Например, запишем ответ, поставив вместо пробела между цифрами знак «-».

Предыдущие решения этой задачи

Номер решения	Время	Размер	Задача	Язык	Результат	Пройдено тестов	Баллы	Посмотреть исходный текст	Просмотреть протокол
10481	2018/10/22 10:09:01	25	2	N/A	Неправильный формат ответа	0	**0**	Просмотр	Просмотр

Рисунок 40 – Неправильный формат ответа

В этом случае нужно просмотреть протокол проверки, в столбце «Доп. информация» будет написано краткое сообщение проверяющей программы о том, почему решение не было принято на проверку. В данном случае указано, что в решении содержится недопустимый символ «-».


Неправильный формат ответа

N	Результат	Баллы	Доп. информация
1	Неправильный формат ответа	0 (0)	Строка номер 1 содержит недопустимый символ: '-'

Рисунок 41 – Недопустимый символ

Необходимо проверить, какой ответ был сдан в тестирующую систему, для этого нужно вернуться на страницу сдачи задачи и нажать на ссылку «Просмотр» в столбце «Просмотреть исходный текст».

Рисунок 42 – Просмотр исходного текста

После исправления ошибок в записи ответа, нужно отправить на проверку новый ответ по данной задаче.

Сообщение тестирующей системы об ошибках в программе может быть различным. Например, если в строке записаны три числа, а не два, сообщение об ошибке будет следующим.


Неправильный формат ответа

N	Результат	Баллы	Доп. информация
1	Неправильный формат ответа	0 (0)	В строке номер 1 должно быть ровно два числа

Рисунок 43 – Сообщение об ошибки

В данной задаче в одной строке должны быть два различных числа, если ввести два равных числа в строке, сообщение об ошибке будет следующим.

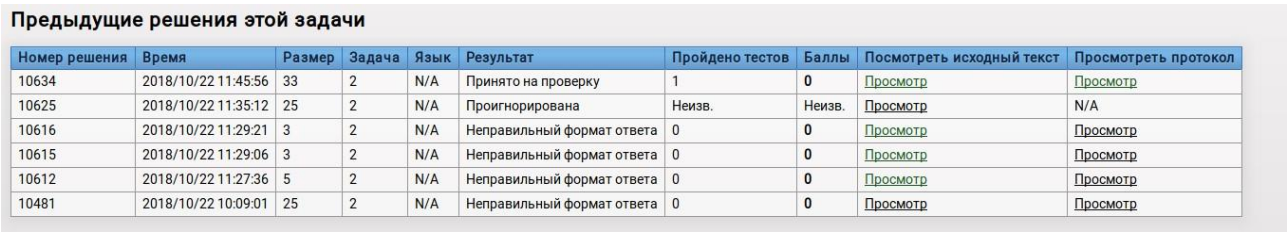
Неправильный формат ответа

N	Результат	Баллы	Доп. информация
1	Неправильный формат ответа	0 (0)	В строке номер 1 записано два одинаковых числа

Рисунок 44 – Сообщение об ошибки при двух равных числах в строке

Если форма записи ответа удовлетворяет условию задачи, то результат тестирования будет «Принято на проверку». Этот статус не означает, что решение полностью или хотя бы частично верное, решение может быть неправильным, но формат записи ответа был соблюдён. Баллы за такие задачи не сообщаются до конца олимпиады.

Участник может сдать несколько ответов по одной задаче, при этом оценивается последнее решение, которое получило статус «Принято на проверку». Если участник сдаёт ещё одно решение, которое было принято на проверку, то у предыдущего принятого на проверку решения статус меняется на «Проигнорирована».



Предыдущие решения этой задачи

Номер решения	Время	Размер	Задача	Язык	Результат	Пройдено тестов	Баллы	Посмотреть исходный текст	Просмотреть протокол
10634	2018/10/22 11:45:56	33	2	N/A	Принято на проверку	1	0	Просмотр	Просмотр
10625	2018/10/22 11:35:12	25	2	N/A	Проигнорирована	Неизв.	Неизв.	Просмотр	N/A
10616	2018/10/22 11:29:21	3	2	N/A	Неправильный формат ответа	0	0	Просмотр	Просмотр
10615	2018/10/22 11:29:06	3	2	N/A	Неправильный формат ответа	0	0	Просмотр	Просмотр
10612	2018/10/22 11:27:36	5	2	N/A	Неправильный формат ответа	0	0	Просмотр	Просмотр
10481	2018/10/22 10:09:01	25	2	N/A	Неправильный формат ответа	0	0	Просмотр	Просмотр

Рисунок 45 – Решение проигнорировано

На изображении выше сначала на проверку были отправлены 4 решения, форма записи ответа которых не соответствовала условию задачи, затем было отправлено решение № 10625, которое было принято на проверку, затем было отправлено ещё одно решение № 10634, которое было принято на проверку, а статус решения № 10634 изменился на «Проигнорирована».

Решения, которые не были приняты на проверку, не будут оценены, для получения баллов по задаче необходимо сдать решение, которое получит статус «Принято на проверку»

при условии, что оно будет правильным).

По ссылке «Итог» отображается информация по всем задачам.

Итог по задачам

Задача	Статус	Пройдено тестов	Баллы	Номер решения
1	Принято на проверку			10637
2	Принято на проверку			10634
3				
4				
5	Неполное решение	7	6	10646
6				
7				

Суммарный балл: 6

Рисунок 46 – Итог по задачам на проверку

В данном случае на проверку были приняты решения по задачам 1 и 2 с записью ответа. Задачи 5, 6, 7 – задачи по программированию, по этим задачам результат проверки и баллы сообщаются сразу же. Суммарный балл выставлен только по задачам по программированию, по задачам с записью ответа баллы не сообщаются до окончания олимпиады.

В олимпиадной задаче по традиции используется задача с формулой. Ниже приведен пример.

Из Калининграда в Ханты-Мансийск отправляется автобус, скорость которого составляет v км/ч. В это же время из Ханты-Мансийска в Калининград стартует улучшенный автобус «Газель», скорость которого составляет w км/ч. Длина маршрута между городами составляет s км. Необходимо определить, какой длины проделает маршрут автобус «Газель», прежде чем встретится с автобусов, который выехал ему на встречу. Автобусы двигаются с постоянной скоростью.

Решение строится таким образом, что ответом будет являться формула, переменные которой записаны латиницей, суммирование отображается знаком плюс, а вычитание знаком минус, умножение обозначается звездочкой, а деление – слешом. Скобки для обозначения порядка выполнения арифметических действий.

Например, опускать знак умножения, как это делается в обычных формулах недопустимо.

Решением такой задачи будет служить верно написанное выражение.

Арифметические операции соответствуют общепринятым. Приоритет операций такой же как в математике. Например первым выполняется умножение нежели сложение. Чтобы изменить порядок действий необходимо использовать скобки. Наличие пробельных символов неважно.

Правильный пример на эту задачу:

w * s / v + w)

Любое выражение, которое в результате получит тот же ответ, будет верное.

s - v * s / w + v)

Такой ответ тоже будет верным s /w / v + (1 * w / v)

Проверка решения осуществляется на языке python. Выражение нужно, чтобы состояло только из переменных скобок и арифметических операций.

При ошибочном вводе, выводится ошибка «Неправильный формат ввода»

Необходимо пересмотреть решение. Ниже приведены наиболее частые ошибки, которые допускаются при решении этой задачи.

Ошибочный ответ	**Какая допущена ошибка в записи**
w*s:v+w)	Неверный знак деления.
ц * s /v + w)	Использование кириллицы.
w * sv + w).	Лишний символ точка
w * s /v + w) км	Лишние символы в конце строки
w *s / v + w)	В выражении одна скобка
w * s / [v + w]	Неверно используются скобки
w s / v + w)	Отсутствует операция умножения
s = w * s / v + w)	Используется равенство
Путь равен w*s/v+w)	Использование недопустимых символов в начале строки

Арифметическое выражение, которое будет состоять только из чисел, пройдет данный тест, но наберет 0 баллов.

В конце олимпиады можно увидеть итог по задачам, который изображен ниже.

Итог по задачам

Задача	Статус	Пройдено тестов	Баллы	Номер решения
1	OK	1	10	10906
2	Неполное решение	0	4	10657
3	Неполное решение	0	0	10638
4				
5	Неполное решение	7	6	10646
6				
7				

Суммарный балл: 20

Рисунок 47 – Баллы итоговые по задачам

На странице «Посылки» приведены данные по результатам всех задач.

Отправленные решения (последние 15)

Номер решения	Время	Размер	Задача	Язык	Результат	Пройдено тестов	Баллы	Посмотреть исходный текст	Просмотреть протокол
10906	2018/10/22 13:02:26	15	1	N/A	OK	1	**10**	Просмотр	Просмотр
10657	2018/10/22 11:59:14	37	2	N/A	Неполное решение	0	**4**	Просмотр	Просмотр
10646	2018/10/22 11:53:27	121	5	python3	Неполное решение	7	**6**	Просмотр	Просмотр
10638	2018/10/22 11:49:14	3	3	N/A	Неполное решение	0	**0**	Просмотр	Просмотр
10637	2018/10/22 11:49:04	3	1	N/A	Проигнорирована	Неизв.	Неизв.	Просмотр	N/A
10634	2018/10/22 11:45:56	33	2	N/A	Проигнорирована	Неизв.	Неизв.	Просмотр	N/A
10625	2018/10/22 11:35:12	25	2	N/A	Проигнорирована	Неизв.	Неизв.	Просмотр	N/A
10616	2018/10/22 11:29:21	3	2	N/A	Неполное решение	0	**0**	Просмотр	Просмотр
10615	2018/10/22 11:29:06	3	2	N/A	Неполное решение	0	**0**	Просмотр	Просмотр
10612	2018/10/22 11:27:36	5	2	N/A	Неполное решение	0	**0**	Просмотр	Просмотр
10481	2018/10/22 10:09:01	25	2	N/A	Неполное решение	0	**0**	Просмотр	Просмотр

Посмотреть всё

Рисунок 48 – Просмотр решенных задач

По умолчанию на этой странице отображены последние 15 отправленных решений, чтобы просмотреть все решения, нужно нажать на ссылку «Просмотреть всё».

На рисунке 44 можно увидеть решения 15-ти задач, которые были отправлены, для того, чтобы увидеть полный список решений нужно нажать на кнопку «Просмотреть всё»

Когда заканчивается олимпиада, то в сводке решений содержится подробная информация о решениях и баллах, которые были за них получены. Для примера взглянем на протокол под сортировке массива.

Неправильный ответ

N	Результат	Баллы	Доп. информация
1	Неправильный ответ	4 (10)	[3, 7, 1, 6, 2, 4, 8, 5] - исходное состояние списка [1, 7, 3, 6, 2, 4, 8, 5] - состояни...

```
====== Тест #1 =======
--- Результат работы: размер 37 ---
1 3
2 7
4 6
5 7
7 8
1 2
1 2


--- Правильный ответ: размер 0 ---

--- Вывод проверяющей программы: размер 776 ---
[3, 7, 1, 6, 2, 4, 8, 5] - исходное состояние списка
[1, 7, 3, 6, 2, 4, 8, 5] - состояние списка после обмена 1
[1, 2, 3, 6, 7, 4, 8, 5] - состояние списка после обмена 2
[1, 2, 3, 4, 7, 6, 8, 5] - состояние списка после обмена 3
[1, 2, 3, 4, 5, 6, 8, 7] - состояние списка после обмена 4
[1, 2, 3, 4, 5, 6, 7, 8] - состояние списка после обмена 5
[2, 1, 3, 4, 5, 6, 7, 8] - состояние списка после обмена 6
[1, 2, 3, 4, 5, 6, 7, 8] - состояние списка после обмена 7
Задача решена за 7 обменов, но есть более короткое решение
```

Рисунок 49 – Проверка по тестам решение задач

Внизу виден вывод пошаговых операций при работе с массивом. В ходе его упорядочивания был получен отсортированный массив. Решение, приведенное на рисунке 45 заняло 7 операций. Лучшее решение содержит меньшее число операций обмена – 5. Оценка за данное решение невысокая и составляет 4 балла из 10.

ЗАКЛЮЧЕНИЕ

В результате исследования был проведен обзор и анализ литературы на заявленную тему, определены объект, предмет, гипотеза, цели и задачи исследования, актуальность данной работы.

В результате исследования было предложено создание тестирующей подсистемы в Ejudge, устраняющие и предоставляющее дополнительную разработку для проведения олимпиад по информатике на муниципальном этапе по Амурской области.

Все поставленные цели были выполнены, подсистема была отлажена и запущена в Oracle VM VirtualBox Fedora-26-32- ejudge в тестовом режиме.

Получены навыки проектирования интерфейса и разработки программного обеспечения на языке программирования C++.

Практическим результатом проделанной работы является создание тестирующей подсистемы. Разработан интерфейс для пользователя и администратора для решения задач, который упростит работу проведение олимпиады Всероссийского этапа школьников по информатике в г. Благовещенске:

1. Сократит количество дней олимпиады, не нужно будет делать пробный тур;
2. Повысит качество обучающихся в проведении муниципального этапа олимпиады;
3. Автоматизированная проверка без участия человека;
4. Изменение таблицы результатов в режиме реального времени;
5. Сократит время подведения итогов.

Среди достоинств разрабатываемой подсистемы стоит отметить возможность ведения протоколов жюри в электронной виде, а также подведение итогов можно подвести удаленно. Минус в этом протокол жюри не смогут подписать пока не будет электронной подписи.

Данную подсистему можно дальше усовершенствовать.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Баженова, И.Ю. Языки программирования: Учебник для студентов учреждений высш. проф. образования / И.Ю. Баженова; Под ред. В.А. Сухомлин. – М.: ИЦ Академия, 2016. – 368 с.
2. Бейли, Л. PHP и MySQL / Линн Бейли; пер. с англ. Н. Котляров. – М.: Питер, 2016. – 592 с.
3. Гуриков, С.Р. Основы алгоритмизации и программирования на Python: Учебное пособие / С.Р. Гуриков. - М.: Форум, 2018. - 384 с.
4. Замятина, О.М. Вычислительные системы, сети и телекоммуникации. моделирование сетей.: Учебное пособие для магистратуры / О.М. Замятина. - Люберцы: Юрайт, 2016. - 159 с.
5. Информационные технологии и вычислительные системы: Обработка информации и анализ данных. Программная инженерия. Математическое моделирование. Прикладные аспекты информатики / Под ред. С.В. Емельянова. - М.: Ленанд, 2015. - 104 с.
6. Информационные технологии и вычислительные системы: Обработка информации и анализ данных. Программная инженерия. Математическое моделирование. Прикладные аспекты информатики / Под ред. С.В. Емельянова. - М.: Ленанд, 2015. - 104 с.
7. Керниган, Б. Язык программирования С. 2-е изд. / Б. Керниган, Д.М. Ритчи. – М.: Вильямс, 2016. – 288 с.
8. Косяков, А. Системная инженерия. Принципы и практика / А. Косяков, У. Свит, С. Сеймур, С. Бимер. - М.: ДМК, 2014. - 624 с.
9. Макин, Дж.К. Проектирование серверной инфраструктуры баз данных Microsoft SQL Server 2005 / Дж.К. Макин. - М.: Русская редакция, 2008. - 560 с.
10. Максимцов, М.М. Исследование систем управления: Учебное пособие / М.М. Максимцов, А.В. Игнатьева. - М.: Юнити, 2017. - 64 с.

11. Малыхина, М.П. Базы данных: основы, проектирование, использование / М.П. Малыхина. - СПб.: BHV, 2007. - 528 с.

12. Мартишин, С.А. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench: Методы и средства проектирования информационных систем и технолог / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. - М.: Форум, 2017. - 62 с.

13. Мартишин, С.А. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench: Методы и средства проектирования информационных систем и техноло / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. - М.: Форум, 2018. - 61 с.

14. Мартишин, С.А. Проектирование и реализация баз данных в СУБД MySQL с использованием MySQL Workbench: Методы и средства проектирования информационных систем и техноло / С.А. Мартишин, В.Л. Симонов, М.В. Храпченко. - М.: Форум, 2018. - 61 с.

15. Михайлова, Э.А. Экономическая оценка инвестиций / Э.А. Михайлова, Л.Н. Орлова. – Рыбинск: РГАТА, 2008. – 176 с.

16. Никсон, Р. Создаем динамические web-сайты с помощью PHP, MySQL, CSS и HTML5. - 4-е изд. – СПб.: Питер, 2016. – 768 с.

17. Орлов, С.А. Программная инженерия. Технологии разработки программного обеспечения. Стандарт третьего поления: Учебник / С.А. Орлов. - СПб.: Питер, 2019. - 224 с.

18. Перлова, О.Н. Проектирование и разработка информационных систем: Учебник / О.Н. Перлова, О.П. Ляпина, А.В. Гусева. - М.: Academia, 2017. - 416 с.

19. Перлова, О.Н. Проектирование и разработка информационных систем: Учебник / О.Н. Перлова. - М.: Академия, 2018. - 272 с.

20. Пьюривал, С. Основы разработки web-приложений / Сэмми Пьюривал; пер. с англ. А.Топлеева. – СПб.: Питер, 2015. – 272 с.

21. Ржеуцкая, С.Ю. Базы данных. Язык SQL: учебное пособие / С.Ю. Ржеуцкая. – Вологда: ВоГТУ, 2010. – 159 с.

22. Роббинс, Дж. HTML5, CSS3 и JavaScript. Исчерпывающее руководство / Дженифкр Роббинс; пер. с англ. М.А. Райтман. – М.: Эксмо, 2014. – 528 с.

23. Смирнов, С.Н. Безопасность систем баз данных / С.Н. Смирнов. - М.: Гелиос АРВ, 2016. - 352 с.40.

24. Федоров, С.Е. Компьютерное моделирование и исследование систем автоматического управления: учебно-методическое пособие для вузов / С.Е. Федоров. - М.: Русайнс, 2018. - 256 с.

25. Фридман, А.Л. Основы объектно-ориентированного программирования на языке Си++ / А.Л. Фридман. – М.: Гор. линия-Телеком, 2012. – 234 с

26. Фризен, И.Г. Основы алгоритмизации и программирования (среда PascalABC.Net): Учебное пособие / И.Г. Фризен. - М.: Форум, 2018. - 784 с.

27. Хомоненко, А.Д. Базы данных: Учебник для высших учебных заведений / А.Д. Хомоненко, М.Г. Мальцев, В.М. Цыганков. – СПб.: КОРОНА-Век, 2009. – 736 с.

28. Чипига, А.Ф. Информационная безопасность автоматизированных систем / А.Ф. Чипига. - М.: Гелиос АРВ, 2015. - 336 с.41.

29. Чошанов, М.А. Инженерия обучающих технологий / М.А. Чошанов. - М.: Бином, 2015. - 239 с.

30. Шаньгин, В.Ф. Информационная безопасность компьютерных систем и сетей: Учебное пособие / В.Ф. Шаньгин. - М.: Форум, 2018. - 256 с.

ПРИЛОЖЕНИЕ А

Библиографический список авторских работ по теме диссертации

1. Золотарёва Г.В. Перспективы развития автоматизированной системы тестирования / Г.В. Золотарева, Н.П. Семичевская // Молодежь XXI века: шаг в будущее: материалы XX региональной науно - практической конференции (23 мая 2019 г., Благовещенск): в 3 томах. – Благовещенск: Издательство Амурского гос. ун-та, 2019. – Т.3. – 206 с.
2. Золотарёва Г.В. Автоматизированная система тестирования в образовании/ Г.В. Золотарева, Н.П. Семичевская //Журнал Вестник науки и образования №22 (76) ноябрь 2019, Часть 1: Москва, 37 с.
3. Золотарёва Г.В. статья [Электронный ресурс]/ Г.В. Золотарёва// Перспективы развития автоматизированной системы тестирования в образовании: материалы конференции – Благовещенск, 2019 – Режим доступа: https://infourok.ru/perspektivi-razvitiyaavtomatizirovannoysistemi-testirovaniya-v-obrazovanii-3751876.html –29.05.2019.

ПРИЛОЖЕНИЕ Б

Сертификат участника научной конференции

СЕРТИФИКАТ

участника

Золотарева Галина Викторовна

XX региональной научно-практической конференции «Молодежь XXI века: шаг в будущее»

Секция Информационные технологии цифровизация современного общества

Председатель совета ректоров вузов Амурской области П.В. Тихончук

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АмГУ 1975

Благовещенск
23 мая 2019 г.

ПРИЛОЖЕНИЕ В

Справка о публикации в журнале

PUBLISHING HOUSE «PROBLEMS OF SCIENCE»
ЦС «OLIMP», ITN 3702681148, PSRN 1123702026524, HTTP://SCIENCEPROBLEMS.RU, INFO@PSN.RU, PH.: +7(910)690-15-09

Вестник науки и образования
СВИДЕТЕЛЬСТВО РОСКОМНАДЗОРА О РЕГИСТРАЦИИ СМИ ПИ № ФС77-50633 ОТ 14 МАРТА 2014 г. ISSN 2312-8089

СПРАВКА
О ПРИНЯТИИ СТАТЬИ К ПУБЛИКАЦИИ
№ ВНО-4161 ОТ «2» ДЕКАБРЯ 2019 г.

НАСТОЯЩАЯ СПРАВКА ВЫДАНА ДЛЯ ПРЕДЪЯВЛЕНИЯ ПО МЕСТУ ТРЕБОВАНИЯ И ПОДТВЕРЖДАЕТ ФАКТ ОФИЦИАЛЬНОЙ ПУБЛИКАЦИИ

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ТЕСТИРОВАНИЯ В ОБРАЗОВАНИИ. / AUTOMATED TESTING SYSTEM IN EDUCATION.

АВТОР(ы)
Золотарёва Галина Викторовна,
Семичевская Наталья Петровна

Статья принята к публикации в журнал № 23 (77), 2019 год.
Территория распространения: Российская Федерация, зарубежные страны.
САЙТ ЖУРНАЛА: HTTP://SCIENTIFICJOURNAL.RU

Заведующая редакцией *Ефимова А.В.*

ПРИЛОЖЕНИЕ Г

Сертификат о публикации в журнале

СЕРИЯ: 10 VNO 01242

ИЗДАТЕЛЬСТВО «ПРОБЛЕМЫ НАУКИ»

ПЕЧАТНЫЙ/ЭЛЕКТРОННЫЙ НАУЧНО-МЕТОДИЧЕСКИЙ ЖУРНАЛ

ВЕСТНИК НАУКИ И ОБРАЗОВАНИЯ

СВИДЕТЕЛЬСТВО ФЕДЕРАЛЬНОЙ СЛУЖБЫ ПО НАДЗОРУ В СФЕРЕ СВЯЗИ, ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И МАССОВЫХ КОММУНИКАЦИЙ (РОСКОМНАДЗОР) ПИ № ФС 77-5063 / ЭЛ № ФС 77 - 58456
МЕЖДУНАРОДНЫЕ СТАНДАРТНЫЕ СЕРИЙНЫЕ НОМЕРА: ISSN PRINT 2312-8089, ISSN ONLINE 2541-7851

СЕРТИФИКАТ

О ПУБЛИКАЦИИ В ЖУРНАЛЕ № 22(76) (ноябрь 2019 г.) СТАТЬИ

АВТОМАТИЗИРОВАННАЯ СИСТЕМА ТЕСТИРОВАНИЯ В ОБРАЗОВАНИИ

ОБЛАДАТЕЛЬ СЕРТИФИКАТА

Золотарёва Галина Викторовна

студент магистратуры;

Семичевская Наталья Петровна

научный руководитель, кандидат технических наук, доцент, кафедра информационных и управляющих систем, Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования Амурский государственный университет, г. Благовещенск

РАСПОЛОЖЕНИЕ СТАТЬИ В АРХИВЕ ЖУРНАЛА
HTTP://SCIENTIFICJOURNAL.RU/GLAVNAYA/ARKHIV.HTML

ГЛАВНЫЙ РЕДАКТОР [illegible] ВАЛЬЦЕВ С.В.

М.П.

28 ноября 2019 года

ИЗДАТЕЛЬСТВО ПРОБЛЕМЫ НАУКИ

ПРИЛОЖЕНИЕ Д

Свидетельство о публикации на сайте infourok.ru

ИНФОУРОК

Свидетельство о рег. СМИ Эл. №ФС77-60625 от 20.01.2015 выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций
Лицензия на осуществление образовательной деятельности № 5201 выдана 02 апреля 2018 г. Департаментом Смоленской области по образованию и науке бессрочно
Положение о редакционной коллегии проекта «Инфоурок», утверждено приказом главного редактора от 28.11.2018 №1

СВИДЕТЕЛЬСТВО

о публикации на сайте infourok.ru

Настоящим подтверждается, что

Золотарёва Галина Викторовна

опубликовал(а) на сайте infourok.ru методическую разработку, которая успешно прошла проверку и получила высокую оценку от эксперта "Инфоурок":

ПЕРСПЕКТИВЫ РАЗВИТИЯАВТОМАТИЗИРОВАННОЙСИСТЕМЫ ТЕСТИРОВАНИЯ В ОБРАЗОВАНИИ

Web-адрес публикации:

https://infourok.ru/perspektivi-razvitiyaavtomatizirovannoysistemi-testirovaniya-v-obrazovanii-3751876.html

Данное Свидетельство выдается бесплатно и только при достижении высоких результатов согласно «Манифесту о качестве «Инфоурок». Проверить подлинность документа, а также посмотреть список достижений и результатов, за которые выдан данный документ, можно по ссылке: infourok.ru/standart

ТАСС ИНФОРМАЦИОННОЕ АГЕНТСТВО РОССИИ

Свидетельство о регистрации в Национальном центре ISSN (присвоен Международный стандартный номер сериального издания: № 2587-8018 от 17.05.2017)

ДОКУМЕНТ ВЫДАН В СООТВЕТСТВИИ С «МАНИФЕСТОМ О КАЧЕСТВЕ «ИНФОУРОК» INFOUROK.RU/STANDART

29.05.2019

БВ48131356

Председатель редакционной коллегии проекта «Инфоурок»

И. В. ЖАБОРОВСКИЙ

INFOUROK.RU

ПРИЛОЖЕНИЕ Е

Справка о внедрении

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
АМУРСКОЙ ОБЛАСТИ
ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
УЧРЕЖДЕНИЕ ДОПОЛНИТЕЛЬНОГО
ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«АМУРСКИЙ ОБЛАСТНОЙ ИНСТИТУТ
РАЗВИТИЯ ОБРАЗОВАНИЯ»
(ГАУ ДПО «АмИРО»)
675005, г. Благовещенск, ул. Северная, 107,
тел / факс (4162) 226-262, 226-260
E-mail: amurippk@yandex.ru ОКПО 41707668
29.06.2020 № 01-16-1251
на № _____ от _____

СПРАВКА О ВНЕДРЕНИИ
результатов диссертационного исследования

Золотарёвой Галины Викторовны
на тему: «Разработка тестирующей подсистемы для автоматической проверки задач ГАУ ДПО «АмИРО» муниципального этапа Всероссийской олимпиады по Амурской области»

ГАУ ДПО «АмИРО», подтверждает, что результаты научного исследования Золотарёвой Г.В. с 9.01.2019 по 20.06.2020 были рассмотрены и одобрены к использованию в образовательном процессе, а также будут рекомендованы для использования на этапе проверке заданий муниципального этапа по информатике.

Магистрант Золотарёва Г.В. создала программу внедрение, которой позволит существенно повысить качество проведения муниципального этапа ВсОШ по информатике в Амурской области. Введение цифровой системы в процесс проведения олимпиад позволит получить объективные результаты и сократить сроки проверки.

И.о ректора Ю.В. Борзунова