

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
Кафедра «Прикладная математика и информатика»

01.03.02 Прикладная математика и информатика

СИСТЕМНОЕ ПРОГРАММИРОВАНИЕ И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

БАКАЛАВРСКАЯ РАБОТА

на тему Разработка математического и программного обеспечения для автоматической проверки заданий при использовании дистанционных технологий обучения

Студент _____ Г. Ю. Юрковец _____

Руководитель _____ А. И. Туищев _____

Допустить к защите

Заведующий кафедрой к.тех.н, доцент, А.В. Очеповский _____

«_____» _____ 2016 г.

Тольятти 2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
Кафедра «Прикладная математика и информатика»

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная
математика и информатика»
_____ А. В. Очеповский

«____» _____ 2016 г.

ЗАДАНИЕ
на выполнение бакалаврской работы

Студент Юрковец Геннадий Юрьевич

1. Тема Разработка математического и программного обеспечения для автоматической проверки заданий при использовании дистанционных технологий обучения
2. Срок сдачи студентом законченной выпускной квалификационной работы 24.06.2016.
3. Исходные данные к выпускной квалификационной работе
 - организация web-доступа к системе через Интернет;
 - ограничение доступа к данным посредством аутентификации, авторизации;
 - количество одновременно работающих пользователей, не менее 30;
 - максимальное время формирования ответа за запрос пользователя, не более 5 сек.;
 - режим работы 24/7/365;
 - разработанное программное обеспечение должно удовлетворять спецификации;

4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов, разделов):

Введение.

Глава 1. Исследование проблемы автоматической проверки заданий при дистанционном обучении.

Глава 2. Реализация системы автоматической проверки заданий.

Глава 3. Развертывание и тестирование системы автоматической проверки заданий.

Заключение.

5. Ориентировочный перечень графического и иллюстративного материала презентация, включающая блок-схемы работы приложения, графики, диаграммы, экранные формы, демонстрирующие работоспособность программного продукта.

6. Дата выдачи задания « 11 » января 2016 г.

Руководитель выпускной
квалификационной работы _____ А. И. Туищев

Задание принял к исполнению _____ Г. Ю. Юрковец

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
Кафедра «Прикладная математика и информатика»

УТВЕРЖДАЮ
Зав.кафедрой «Прикладная
математика и информатика»
_____ А.В.Очеповский

«_____» _____ 2016 г.

КАЛЕНДАРНЫЙ ПЛАН
выполнения бакалаврской работы

Студента Юрковец Геннадия Юрьевича

по теме Разработка математического и программного обеспечения для автоматической проверки заданий при использовании дистанционных технологий обучения

Наименование раздела работы	Плановый срок выполнения раздела	Фактический срок выполнения раздела	Отметка о выполнении	Подпись руководителя
Анализ состояния вопроса	1.03.2016	1.03.2016	выполнено	
Разработка обеспечения	11.03.2016	11.03.2016	выполнено	
Программная реализация предложенных решений	25.03.2016	25.03.2016	выполнено	
Апробация предложенных	3.04.2016	3.04.2016	выполнено	

решений на реальных данных				
Написание 3 главы	6.04.2016	6.04.2016	выполнено	
Создание презентационного материала	21.04.2016	21.04.2016	выполнено	
Проверка на наличие заимствований (плагиата) в системе antiplagiat.ru	17.06.2016	17.06.2016	выполнено	
Предварительная защита	6.06.2016- 11.06.2016	31.05.2016-	выполнено	
Сдача на кафедру отзыва научного руководителя и ознакомление с ним	20.06.2016	20.06.2016	выполнено	
Сдача на кафедру комплекта документов для защиты	24.06.2016	24.06.2016	выполнено	
Защита ВКР	27-30.06.2016	29.06.2016	выполнено	

Руководитель выпускной
квалификационной работы

А. И. Туищев

Задание принял к исполнению

Г. Ю. Юрковец

АННОТАЦИЯ

к бакалаврской работе на тему «Разработка математического и программного обеспечения для автоматической проверки заданий при использовании дистанционных технологий обучения»

Бакалаврская работа выполнена студентом Тольяттинского государственного университета Юрковцом Геннадием Юрьевичем.

Объект исследования - процесс обучения с использованием дистанционных образовательных технологий.

Целью данной работы является автоматизация проверки заданий при использовании дистанционных технологий.

Для достижения указанной цели необходимо решение следующих задач:

Теоретические задачи:

1. Исследование проблемы автоматической проверки заданий при дистанционном обучении.
2. Изучение общей информации по реализации дистанционного обучения.
3. Рассмотрение этапов обучения, существующий процесс отправки и проверки заданий.

Проектные задачи:

1. Построение модели новой технологии.
2. Разработка системы автоматической проверки заданий.
3. Тестирование разработанной системы и описание процесса развертывания.

Для решения поставленных задач в выпускной квалификационной работы использовались следующие методы: анализ, синтез, сравнение и обработка исследуемых источников, практический метод.

Во введении раскрывается актуальность исследования по выбранному направлению, ставится проблема, цель и задачи исследования, определяются объект исследования, ставятся цель и задачи, указывается новизна работы.

В главе первой предложено теоретическое обоснование проблем использования дистанционного обучения и систем автоматизированного контроля заданий, сформулированы требования к новой технологии контроля.

В главе второй раскрываются этапы построения модели новой технологии, с учетом предъявляемых к ней требований, то есть построение модели «Как должно быть».

Третья глава посвящена вопросам развертывания и тестирования системы автоматизированной проверки заданий, на основе предъявляемых требований.

Заключение посвящено основным выводам и предложениям по включению автоматизированной системы контроля заданий в дистанционном обучении.

Результаты данного исследования могут быть использованы в системах дистанционного обучения.

Выпускная квалификационная работа содержит пояснительную записку объемом 80 страниц, включая 21 иллюстрацию, введение, три главы, заключение, список литературы из 23 наименований, приложение.

Оглавление

Введение.....	10
Глава 1 Исследование проблемы автоматической проверки заданий при дистанционном обучении.....	13
1.1 Роль технологий дистанционного обучения для современного общества	13
1.1.1 Понятие дистанционного обучения	13
1.1.2 Особенности дистанционного обучения	19
1.1.3 Методика применения дистанционного обучения	20
1.1.4 Система проверки знаний	21
1.2 Анализ применяемых технологии проверки заданий при дистанционном обучении.....	22
1.2.1 Модель «Как есть» типового процесса проверки заданий при использовании дистанционных технологий.....	23
1.3 Формирование требований к новой технологии проверки заданий при дистанционном обучении.....	25
1.3.1 Постановка задания и дальнейшая автоматическая проверка	25
1.3.2 Автоматизация процесса проверки заданий по программированию	27
1.3.3 Автоматизация проверки заданий, выполненных в офисных приложениях.....	30
1.3.4 Требования к разрабатываемой системе.....	32
Глава 2 Реализация системы автоматической проверки заданий	34
2.1 Построение модели новой технологии.....	34
2.1.1 Модель «Как должно быть»	34
2.2 Проектирование базы данных	37
2.2.1 Концептуальное проектирование	37
2.2.2 Логическая модель базы данных	38
2.3 Физическое проектирование базы данных	41
2.3.1 Обоснование СУБД.....	41

2.3.2	Физическая модель базы данных	41
2.4	Реализация серверной части.....	43
Глава 3	Развертывание и тестирование системы автоматической проверки заданий	46
3.1	Тестирование разработанной системы и описание процессов	46
3.2	Развертывание системы	51
	Заключение	53
	Список используемой литературы	54
	Приложение А Перечень тестов для дальнейшей проверки программ.....	57
	Приложение Б Листинг кода.....	64

Введение

Информационные и интернет технологии сегодня активно используются в сфере образования, основой которой в основном являются очная и заочная формы обучения. Однако наряду с традиционными формами обучения широко развивается область образования связанная с дистанционной формой обучения. Интернет как всемирная сеть предоставляет огромные возможности для применения дистанционных технологий в образовательных учреждениях. Распространенность, надежность, относительная дешевизна всемирной сети дает широкие возможности для организации дистанционного обучения, при этом интернет-сервисы предоставляют как возможность установки, так и систему поддержки дистанционного обучения. В образовательной деятельности, в основе которой лежит дистанционное обучение используются как традиционные, так и инновационные формы, методы и средства обучения, основанные на телекоммуникационных и компьютерных технологиях. Таким образом, новые информационные технологии, средствами которых являются персональные компьютеры, информационные сети, интернет-сервисы используются в системах дистанционного обучения в учреждениях сферы образования.

Компьютерные интернет - технологии активно входят в учебную деятельность образовательных учреждений и оказывают влияние на повышение качества подготовки студентов и являются важной частью современного образования.

Дистанционное обучения (ДО) предоставляет ряд плюсов для разных участников образовательной деятельности.

Обучающимся:

- доступность обучения для желающих получить образование вне зависимости от места проживания и удаленности от образовательного учреждения;
- электронные тесты как система оценки знаний более объективна и менее независима от преподавателя.

Преподавателям:

- автоматизацию системы оценки знаний и умений;
- использование мультимедийных образовательных технологий и цифровых образовательных ресурсов.

Образовательным учреждениям:

- повышение рейтинга учреждения и количества студентов;
- экономию на использование в образовательной деятельности дорогостоящих мультимедиа проекторов и другого медийного оборудования.

Процесс дистанционного обучения в вузах построен следующим образом, студент изучает теоретический курс, размещенный на сайте, проходит тестирование, в результате которого получает баллы и допуск к следующему разделу. Тестирование посредством выбора правильного варианта ответа из предложенных недостаточно для адекватной оценки практических знаний и умений студента. Студент также выполняет практические задания и отправляет на проверку преподавателю. При проверке практических заданий возникают проблемы, связанные с большими временными затратами, особенно если эти задания по дисциплине программирование.

Актуальность работы заключается в том, что разработанная система позволит адекватно и стандартизированно проверять задания по программированию и сокращать временные затраты на данный процесс.

Новизна – задача автоматической проверки заданий при использовании дистанционных технологий обучения решается комплексно впервые.

Объект – процесс обучения с использованием дистанционных образовательных технологий.

Целью данной работы является автоматизация проверки заданий при использовании дистанционных технологий.

Задачи, рассматриваемые в данной работе:

Теоретические задачи:

1. Исследование проблемы автоматической проверки заданий при дистанционном обучении.
2. Изучение общей информации по реализации дистанционного обучения.
3. Рассмотрение этапов обучения, существующий процесс отправки и проверки заданий.

Проектные задачи:

1. Построение модели новой технологии.
2. Разработка системы автоматической проверки заданий.
3. Тестирование разработанной системы и описание процесса развертывания.

Выпускная квалификационная работа состоит из:

Первая глава посвящена исследованию проблемы автоматической проверки заданий при дистанционном обучении.

Во второй главе рассматривается реализация системы автоматической проверки заданий.

Третья глава описывает развертывание и тестирование системы автоматической проверки заданий.

Заключение посвящено основным выводам и предложениям по включению автоматизированной системы контроля заданий в дистанционном обучении.

Глава 1 Исследование проблемы автоматической проверки заданий при дистанционном обучении

1.1 Роль технологий дистанционного обучения для современного общества

1.1.1 Понятие дистанционного обучения

Обучение в системах образования это не только процесс получения знаний, умений и навыков, но формирование компетенций обучающихся в различных сферах практической деятельности. Процесс, имеющий конечную цель, достижение которой осуществляется благодаря правильно организованной системе занятий. Результатом процесса обучения и развития личности является образование.

Основными способами получения образования являются очная и заочная формы. Дистанционное обучение (ДО) основывается на компьютерных технологиях и включает в себя лучшие традиционные методы и формы обучения студентов.

Современный образовательный процесс ДО строиться следующим образом. Студент, находясь в удобном для себя месте, выполняет самостоятельную работу по освоению материала и выполнению практических задания, имея при себе комплект специальных средств обучения. В процессе обучения студент проходит специально разработанные тесты. Проверка заданий осуществляется преподавателем, оценку тестирования, как правило, проводит специализированная система. Для студента является возможным получение консультации от преподавателя по телефону, электронной почте, в специализированной системе, построенной по принципу вопрос-ответ, очно.

В основе процесса ДО лежит интерактивное взаимодействие преподавателя и студента между собой и со средствами обучения, причем их пространственное расположение не имеет влияния на процесс обучения. Схема взаимодействия студента через Интернет-сервисы продемонстрирована на рисунке 1.1.



Рисунок 1.1 – Схема взаимодействия студента через Интернет сервисы

В процессе ДО соблюдается принцип гуманистичности, согласно которому студент не должен быть лишен возможности проходить обучение по причине географической или временной изолированности, отсутствием возможности посещать занятия в силу физических недостатков или постоянной занятости на рабочем месте. ДО является следствием процесса информатизации общества и образования, дистанционное обучение включает лучшие элементы других форм обучения. Дистанционное обучение в XXI веке – передовая, гуманистическая, интегрированная форма получения образования.

В процессе организации дистанционного обучения используются программное, учебно-методическое, техническое обеспечение и инструментальные средства.

Инструментальные средства применяются с целью предоставления

студенту теоретических и практических материалов в информационно-образовательной среде.

Учебно-методическое обеспечение включает в себя:

- методики дистанционного обучения;
- база учебных и практических материалов для студента;
- система управления базой материалов для преподавателя;
- промежуточные и итоговые тесты.

Программное обеспечение:

- системные программы;
- прикладные программные комплексы;
- специализированные системы для создания обучающих программ.

Техническое обеспечение включает в себя:

- периферийное оборудование;
- вычислительное оборудование;
- офисное оборудование;
- канал передачи данных.

Понятие дистанционного обучения может трактоваться по-разному, благодаря этому, отражается многообразие подходов понимания данной области.

В процессе дистанционного обучения используются информационные технологии и системы автоматического тестирования, благодаря этому данная модель обучения успешно сочетает в себе элементы очного, заочного обучения, искореняя изъяны традиционных форм обучения, и сохраняя все их достоинства.

Дистанционное обучение реализуется посредством специализированной информационно-образовательной системы, представляет собой комплекс образовательных услуг для широкого слоя населения и зависит от места расположения. Информационно-образовательная среда дистанционного обучения представляет собой систему, реализующую совокупность средств:

- информационные ресурсы;
- передача данных;
- аппаратно-программное обеспечение;
- протоколы взаимодействия;
- организационно-методическое обеспечение.

Система дистанционного обучения построена так, чтобы удовлетворять образовательные потребности пользователей.

ДО реализует право личности на получение образования и доступ к информации, кроме того это одна из форм непрерывного образования.

Дистанционного обучения – это обучение на расстоянии, при этом процесс дистанционного обучения исключает необходимость:

- посещать учебное заведение;
- придерживаться строгого расписания занятий;
- выполнять курс в установленном порядке.

В процессе дистанционного обучения студент сам организует свой учебный процесс, тем самым формирует свой индивидуальный маршрут обучения. В результате чего студент сам выбирает темп, в котором будет изучать теоретический курс и выполнять практические задания. Современное дистанционное обучение реализуется с помощью технологий представленных на рисунке 1.2.

Получать дистанционное образование можно самостоятельно и в on-line классе. Студент самостоятельно задает временные рамки своего обучения, выбирает место для обучения в условиях дома или в центре доступа к Интернет ресурсам. Дистанционное обучение можно считать эффективным, если правильно подобраны технологии, методы, средства обучения, задания. При этом особое внимание стоит уделить желаниям и возможностям студента и преподавателя. Высоких результатов позволяет добиться проведение преподавателем своевременных on-line консультаций для студента.

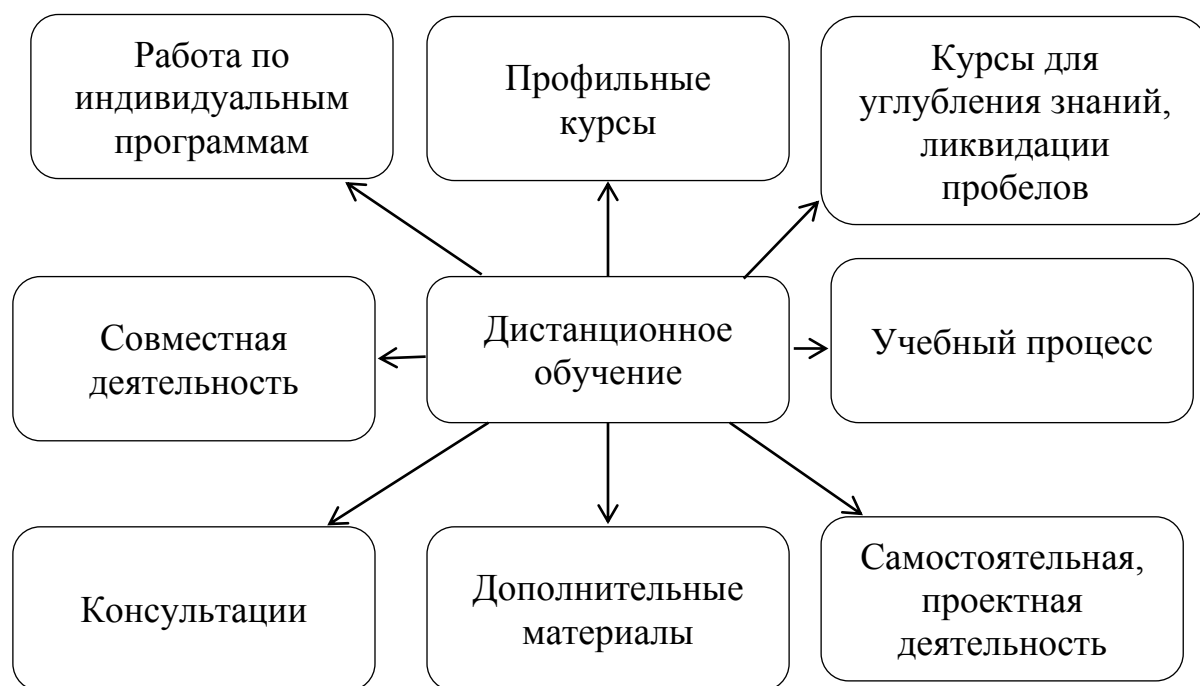


Рисунок 1.2 – Технологии реализации дистанционного обучения

Повышение эффективности дистанционного обучения опирается на наиболее полное и точное согласование требований и возможностей студента. Необходимо учитывать все ограничения, с которыми могут столкнуться преподаватель и студент в процессе обучения. Как правило, в процессе обучения нужно использовать несколько средств общения одновременно, при этом студент не только обучается, но и получает информацию о технических и коммуникационных новшествах. Эффективность дистанционного обучения основана на том, что студенты сами выбирают индивидуальный маршрут и самостоятельно определяют уровень и темп обучения без внешнего давления. Эффективность ДО напрямую зависит от того, сколько времени и сил студент тратит на процесс. Важным аспектом системы дистанционного обучения является наличие полной отчетности студента перед преподавателем. После прохождения каждого раздела курса студент должен решить практические задания и переслать их на проверку преподавателю, только после этого студент может перейти к следующему этапу курса, данный процесс отражен на рисунке 1.3.

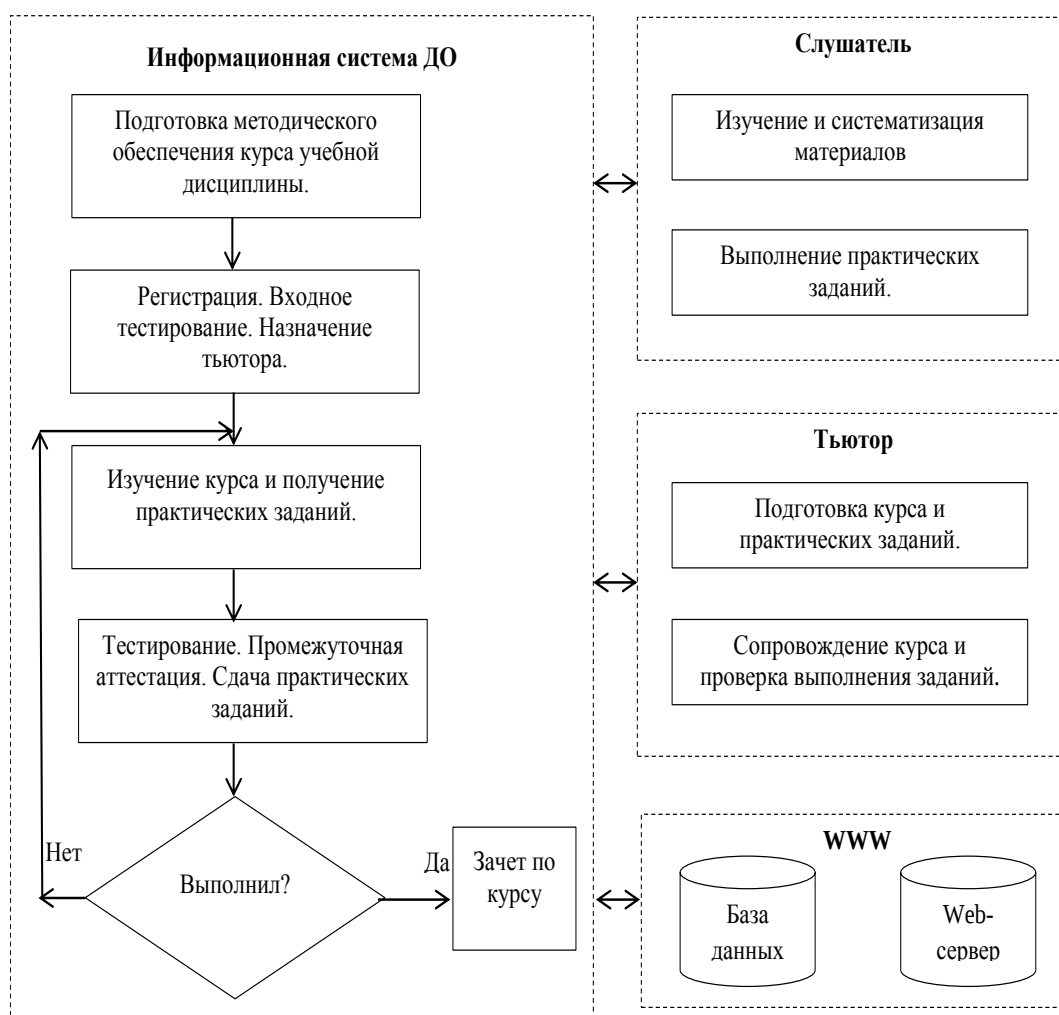


Рисунок 1.3 – Укрупненная схема взаимодействия в системе дистанционного обучения

Однако при всех достоинствах, система дистанционного обучения уязвима. Самым сложным этапом является идентификация студента - проверить, кто именно выполнил и сдал задание, пока невозможно. Учебные заведения использующие систему ДО как правило вводят обязательную очную сессию, студент обязан явиться на экзамен, предъявить документ, подтверждающий личность, после этого он допускается к экзамену. Кроме того в системе ДО, теряется лично-очный контакт преподавателя и студента.

1.1.2 Особенности дистанционного обучения

В процессе анализа теории и практики использования дистанционного обучения (ДО) можно выделить существенные особенности, характеризующие ДО. Рассмотрим их.

Гибкость. Студенты, сами выбирают удобное для себя время, место и темп для обучения. Студент может учиться столько, сколько считает необходимым для освоения курса и получения необходимых знаний по изучаемым дисциплинам.

Модульность. Системы дистанционного обучения основаны на модульном принципе. Отдельно взятый учебный курс соответствует изучаемой предметной области, что позволяет, учитывая индивидуальные или групповые потребности, сформировать учебный план из наборов независимых учебных курсов.

Параллельность. Студент может совмещать получение образования и профессиональную деятельность.

Дальнодействие. Нахождение студента на расстоянии от образовательного учреждения, не влияет на эффективность образовательного процесса.

Асинхронность. В процессе обучения преподаватель и студент работают по удобному и независимому расписанию.

Охват. Эта особенность подразумевает «массовость». Процесс обучения никак не зависит от количества студентов.

Рентабельность. Эта особенность представляет собой экономическую эффективность дистанционного обучения.

Преподаватель. У преподавателя в процессе ДО, появляются новые задачи и функции.

Студент. Требования, предъявляемые к студенту, отличаются от общепринятых.

Новые информационные технологии. В дистанционном обучении применяются все виды информационных технологий, одним из важнейших аспектов является автоматизация процесса выдачи и проверки заданий.

Представленные выше особенности дают преимущества дистанционному обучению перед другими формами получения образования, но, одновременно определяют характерные требования к преподавателю и к студенту.

1.1.3 Методика применения дистанционного обучения

Образовательные учреждения высшего, среднего и дополнительного профессионального образования используют в своей деятельности разработанные методики использования дистанционного образования и технологий. Исходя из этого, определяются основные требования, которые предъявляются к программам, применяемым в дистанционном обучении. Далее рассматриваются основные аспекты методики использования дистанционного образования, приведенные в данной работе.

Дистанционное обучение реализуется использованием системы технологий обучения, которые позволяют осуществить взаимодействие обучающегося и преподавателя независимо от места их расположения и времени нахождения в информационной системе дистанционного обучения на основе информационных технологий. В данный момент основной дистанционной образовательной технологией является Интернет-технология.

Целью дистанционного обучения является предоставление обучающимся возможности освоения основных или дополнительных профессиональных образовательных программ как высшего, так и среднего профессионального образования по месту жительства.

Учебное заведение может организовывать образовательную деятельность дистанционного обучения, используя следующие формы:

- очную;
- очно-заочную (вечернюю);

- заочную;
- сочетание выше указанных форм.

В процессе дистанционного обучения наряду с традиционными информационными технологиями обучения используют следующие средства:

- мультимедийные учебные пособия;
- электронные учебники;
- компьютерные программы (тренинги);
- учебные пособия;
- тестирующие комплекты;
- цифровые лабораторные практикумы;
- аудиозаписи;
- учебные видеофильмы,
- иные материалы.

Образовательное учреждение реализует текущий контроль и промежуточную аттестацию обучающихся традиционными методами или с использованием информационных систем, которые включают в себя идентификацию личности студента и допуск к электронному тестированию. Обязательную итоговую аттестацию студентов выпускных курсов осуществляют традиционными методами.

1.1.4 Система проверки знаний

Современное образование и, в особенности, дистанционное обучение невозможно представить без применения автоматизированной системы, которая оценивает уровень подготовки студента. Современный образовательный процесс содержит в себе систему тестирования с автоматической обработкой результатов, организующую тестирование посредством web-сайта. Студент, используя логин и пароль, авторизуется на сайте, выбирает тест по дисциплине, проходит данный тест, результаты теста автоматически обрабатываются на сервере и студент получает результат.

Результаты тестирования заносятся в базу данных, это позволяет преподавателю видеть статистику всех студентов.

Системы тестирования имеют ограничения по оценке знаний путем выбора ответа из нескольких вариантов ответов. Соответственно существует необходимость в разработке систем, обеспечивающих автоматическую проверку практических заданий, например задач по программированию.

1.2 Анализ применяемых технологии проверки заданий при дистанционном обучении

Существующие системы образования имеют основные дидактические процедуры учебной деятельности и варианты их реализации [13]:

1. Отбор и систематизация знаний.
2. Формализация знаний в зависимости от учебного плана, подготовка учебных пособий.
3. Доставка знаний к учащимся.
4. Усвоение новых знаний.
5. Тренинг, практическое использование полученных знаний в моделируемых ситуациях.
6. Оценка знаний, аттестация.

Рассмотрим последний вариант (этап оценки знаний) более подробно.

Проблема контроля знаний как многоаспектная область имеет много нерешенных вопросов. Дистанционное обучение, сохраняет эти вопросы, и рассматривает их в новом виде.

Контроль в системе дистанционного обучения можно рассмотреть с разных точек зрения и подходов.

Первая точка зрения – это контроль активности студента. Данный подход применяется преподавателями для зачета курса студенту в зависимости от количества посещенных лекций, практик и отработанных лабораторных работ. Подход является простым в реализации, так как фиксируется посещение студентом личного кабинета в системе дистанционного обучения, что

соответственно определяет активность в обучении. Однако недостаток такого подхода в отсутствии гарантии получения студентом знаний, умений и навыков.

Вторая точка зрения выявляет компетентность студента по изучаемому предмету, то есть, как хорошо он умеет применить знания для реализации теоретических знаний при решении практических заданий. Именно такой подход к контролю знаний наиболее часто используется в системе дистанционного обучения. Данный подход, выявляет вопрос проверки компетентности студента, на который можно ответить с трех позиций:

- компетентность - теоретические знания;
- компетентность - применение знаний на практике;
- компетентность – использование знаний и умений в конкретной профессиональной области.

Все эти позиции имеют как достоинства, так и недостатки, поэтому в практике дистанционного обучения рассматривают совокупность данных позиций.

Рассмотрим существующую на сегодняшний день модель реализации дистанционного обучения с точки зрения контроля выполненных заданий студентом.

1.2.1 Модель «Как есть» типового процесса проверки заданий при использовании дистанционных технологий

В данный момент времени процесс дистанционного обучения построен так, основным методом проверки знаний и подведения промежуточной аттестации, является тестирование. Применяемый вид тестирования, как правило, построен по средствам информационной системы, включающей в себя модули:

- авторизации студента и преподавателя;

- модуль тестирования для студента (автоматическая проверка выбранных вариантов ответа);
- модуль составления тестов для преподавателя (администратора);
- модуль вывода результатов тестирования.

Также проверка знаний студента проводится путем выполнения и сдачи практических заданий преподавателю. Студент посредством информационной системы (портала), электронной почты, получает практическое задание от преподавателя, выполняет его, и высылает на проверку преподавателю. Преподаватель, выполнив проверку и оценку практических заданий, выставляет баллы студентов на портал. На рисунке 1.4 представлена декомпозиция процесса проверки знаний.

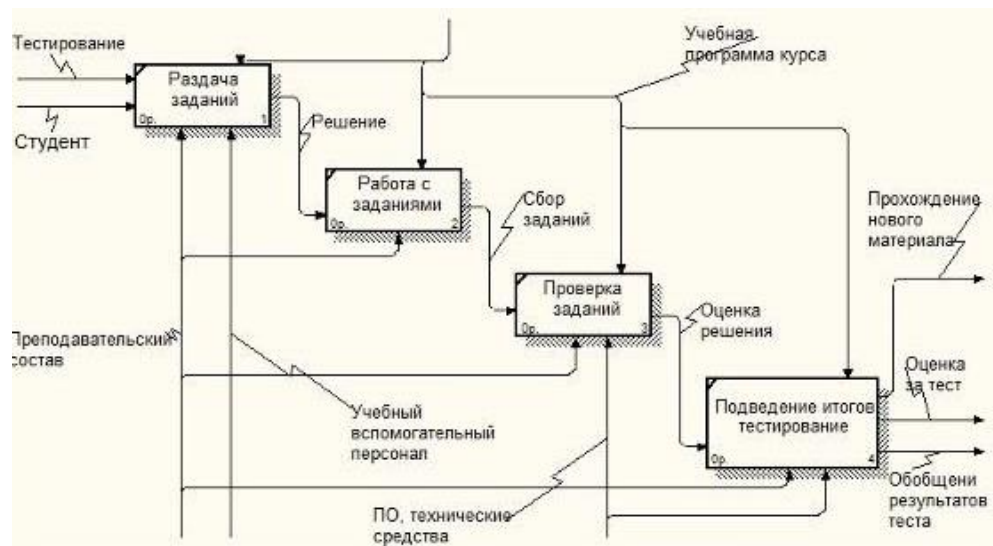


Рисунок 1.4 – Декомпозиция проверки знаний

Недостатками данной схемы взаимодействия преподавателя и студента в системе дистанционного обучения являются:

- тестирование не дает адекватной оценки знаний студента;
- проверка практических заданий занимает много времени у преподавателя.

Мы будем рассматривать данный процесс на примере применения дистанционных технологий обучения к дисциплинам, связанным с информационными технологиями и программированием.

При проверке практических заданий по программированию перед преподавателем стоит несколько проблем:

- для полноценной проверки программ необходимо вручную проверить их на большом наборе тестов (наборе входных данных);
- очень большие временные затраты при полноценной проверке программ;
- при некорректном составлении условия для выполнения заданий, версии программ или компонентов у преподавателя и студента могут не сходиться, что приведет к некорректной работе программы или полной неработоспособности.

1.3 Формирование требований к новой технологии проверки заданий при дистанционном обучении

1.3.1 Постановка задания и дальнейшая автоматическая проверка

Учитывая особенности учебных программ должны быть сформулированы, дополнены и модифицированы принципы. Так тестирование программ должно проводиться, опираясь на четкий план испытаний, этот процесс должен объективно оценивать задания, решая проблемы, описанные в пункте 1.2.1.

Перед тем как перейти к тестированию необходимо тщательно подобрать данные для контрольных примеров, само тестирование происходит последовательно с промежуточной записью результатов. Тестирование заданий направлено на оценку правильности работы программы во всех возможных практических ситуациях, заранее описанных в условии, а также на оценку ее эффективности. Для преподавателя задание решённое учащимся является «черным ящиком». Проверку такой программы без участия студента, возможно, провести при условии четкой и полной постановки условия.

Особое внимание при этом следует уделять [23]:

- форматам входных и выходных данных;

- диапазонам допустимых значений;
- типам входных величин.

Обязательно надо указывать тестовые примеры входных и выходных данных в условии. Благодаря этому студент сможет проверить, правильность понимания им задания. В данный момент обязательной частью постановки условия учебного задания по программированию является четкое указание максимального допустимого времени работы для любых исходных данных, непротиворечащих условию. Благодаря этому указанию в дальнейшем оценивается эффективность алгоритма используемого студентом в процессе написания программы.

Приведем пример правильной и неправильной постановки условия задания с точки зрения ее дальнейшего автоматического тестирования.

Задание 1. Написать программу сортировки числового массива по не убыванию.

Задание 2. Написать программу сортировки массива, по не убыванию. Массив состоит из целых чисел, каждое число по модулю не больше 32 000.

Для ввода входных данных используется клавиатура. В первую строку вводится размер массива — число N , следующие строки представляют собой N элементов массива, для разделения элементов используются пробелы и/или символы переноса строки. Полученные в результате работы программы — N отсортированных чисел, вывести в текстовый файл RESULT.TXT, для разделения отсортированных чисел использовать пробелы и/или символы переноса строки. На одном тесте максимальное время работы программы не должно превышать 2 секунд.

Пример входных данных:

4

6 1

4

3

Результат записи в файл RESULT.TXT для данного набора входных данных:

1 3 4 6

Таким образом, задание 2 более полно и конкретизировано описывает условие нахождения результата.

1.3.2 Автоматизация процесса проверки заданий по программированию

Самым сложным и важным этапом проверки заданий по программированию является автоматизация процесса проверки программы, которая позволяет в полном объеме провести тестирование. Автоматическое тестирование реализуется с помощью программы, выполняющей определенный список действий:

1. Получает данные из очередного теста.
2. Отправляет их во входной файл или на вход тестируемой программы.
3. Получает выходные данные по окончании работы программы.
4. Записывает полученные данные в файл.
5. Проверяет полученный файл.

Тестирующая система должна прерывать выполнение программы студента по истечении времени отведенного на ее работу, записывать и выводить список ошибок возникающих в процессе проверки программы на всех тестах.

Для правильной проверки работы программы на наборе тестов необходимо составить набор тестовых данных, руководствуясь пунктами сформулированными ранее. Данные записываются в файлы с соответствующими именами. Последние символы в имени файла указывают на его порядковый номер в системе тестов. При правильной организации тестирования в процессе возрастания порядковых номеров, возрастает сложность тестов. Для того чтобы система тестирования начала

функционировать необходимо составить конфигурационный файл. Необходимая информация, которая должна содержаться в конфигурационном файле, представлена ниже [9]:

1. Наименование задачи.
2. Способ ввода и вывода данных проверяемой программой (консоль или файл, в последнем случае указывается еще и имя входного и выходного файлов согласно условию задачи).
3. Количество тестов (при отсутствии данной информации осуществляется проверка программы на всех тестах, находящихся в текущем каталоге).
4. Шаблон для имен тестовых входных файлов.
5. Шаблон для имен файлов правильных ответов (иногда, например, в случае неоднозначных правильных ответов, такие файлы могут отсутствовать).
6. Программа для проверки правильности полученного ответа (возможно, как указание одной из стандартных программ, так и созданной специально для проверки данной задачи).
7. Время работы проверяемой программы в секундах на одном тесте.

Система тестирования организует несколько алгоритмов направленных на оценку правильности полученных ответов. Один из алгоритмов это так называемый компаратор – его задача сравнить два файла и выявить совпадения. В данном случае сравнивается файл содержащий данные с вывода проверяемой программы, и файл с заранее подготовленным правильным ответом на данный тест. Для получения файла с верными ответами необходимо реализовать эталонную программу реализующую решение исходной задачи.

Способов анализа полученных результатов огромное количество в связи с этим невозможно предусмотреть все исходы. Это связано с тем, что в ряде задач возможно использование множества вариантов с правильным ответом. Для решения данной проблемы система тестирования в ряде случаев требует создать и реализовать алгоритм для проверки правильности результатов,

который по сложности не уступает решаемой задаче. В некоторых случаях задача реализованного алгоритма сводиться к реверсивной проверке, алгоритм осуществляет подстановку ответов в исходную систему, примером служат системы уравнений и системы шифрования данных. Использование данного метода позволяет проводить тестирование без наличия эталонной программы. При тестировании программы, которая реализует нахождение кратчайшего пути из лабиринта, стоит учитывать, что найденный путь может не совпадать с путем, найденным в результате работы эталонной программы, необходимо проверить, что предложенный путь существует в рамках заданного лабиринта и является кратчайшим. Для проверки условия, является ли предложенный путь кратчайшим, необходимо сравнить его длину с имеющимся правильным ответом.

Если проверка ответов на корректность не вызывает проблем, то любая программа может быть автоматически протестирована, с тем условием что для данной задачи имеется заранее составленная система тестов. Во-первых, текст программы анализируется на безопасность (например, нельзя использовать расширенную память, защищенный режим процессора, считывать и запоминать векторы прерываний, использовать сетевые средства, создавать директории и файлы, которые отсутствуют в условии задачи), а затем скомпилирован. Расширение файла с решением определяет тип компилятора.

Если не удалось успешно осуществить компиляцию, то система проверки в качестве ошибки выведет *CompilationError*. Если все прошло успешно, то создается исполняемый файл для проверяемой программы, далее он последовательно запускается на всем наборе тестов для конкретной задачи. Если после завершения работы программы на тесте ее код равен нулю, необходимо запустить программу проверки правильности выходных данных. В результате успешного тестирования получаем сообщение о правильности работы программы, при неуспешном исходе получаем одну из описанных ранние ошибок. Результаты тестирования пишутся в файл, который доступен для просмотра после окончания тестирования. Также при выводе результатов

тестирования необходимо указать набор входных данных, на котором в работе программы была выявлена ошибка.

Дальнейшая автоматизация может быть достигнута путем использования командных файлов для сбора программ студентов в текущий каталог, и дальнейшую их проверку в порядке очереди. Автоматизация тестирования должна предполагать использование современных средств сетевой передачи файлов, и отображение результатов проверки для студента в любое время посредством сети.

Описанная автоматическая система тестирования имеет открытую архитектуру, благодаря чему возможно подключить неограниченное число компиляторов, а также составить произвольный расширяемый список заданий.

1.3.3 Автоматизация проверки заданий, выполненных в офисных приложениях

Особое внимание некоторых специалистов посвящено решению проблемы автоматизации контроля знаний при изучении дисциплин и курсов. Способов решения данной проблемы множество. Самое простое это организовать процесс компьютерного тестирования, который был рассмотрен ранее. Также можно использовать компьютерные средства контроля практических навыков и умений, приобретенных студентом в процессе обучения. В процессе современного дистанционного образования чаще всего применяются различные виды тестирования, которые позволяют эффективно контролировать низкий уровень усвоения знаний – уровень ознакомления. Это обусловлено тем, что в процессе использования систем тестирования с выбором правильного варианта ответа, или группы ответов, деятельность студента ограничена мыслительными операциями, которые связаны с воспроизведением или классификацией. Объективная оценка практических знаний и умений студента не может основываться на применении одной технологии тестирования. Данная проблема заключается в том, что выбор правильного варианта ответа из списка не является показателем практических

умений студента. Так при изучении прикладного пакета MS Office в курсе обучения, лучший способ проверить знания полученные студентом, это оценить правильность составления документа, созданного в одном из приложений пакета.

Данная работа рассматривает разработку системы, реализующую автоматическую проверку практических заданий (навыков и умений), приобретенных в процессе изучения студентом курса по офисным приложениям. Разрабатываемая система проверяет практические задания и умения в соответствии с заданными требованиями и выявляет степень соответствия результатов. Требования необходимые для автоматической проверки практических заданий:

- составление документа заданной структуры;
- выполнение обязательных элементов в документе;
- соответствие структуры документа заданным параметрам и значениям.

Результат выполненного студентом задания – это документ, созданный в офисном приложении. Любой документ, созданный посредством прикладного пакета MS Office, является сложноорганизованной структурой данных. Данная структура представляет собой набор элементов, составляющих содержание документа, и параметры, заданные для этого набора элементов. Документ MS Word представляет собой совокупность текстовых абзацев и внедренных нетекстовых элементов:

- диаграмм;
- таблиц;
- рисунков.

Каждому абзацу задаётся набор параметров (отступы, выравнивание и т. п.), шрифты и внедрённые элементы также имеют свои наборы параметров. Документ MS Excel – имеет структуру, состоящую из:

- листов;

- ячеек таблиц, в которых содержатся данные определенного типа (формулы, числа и строки);
- диаграмм и т. п.

Таким образом, анализируя структуру созданного студентом MS - документа и параметры его элементов можно осуществить контроль практических навыков и умений студента.

1.3.4 Требования к разрабатываемой системе

Из проведенного анализа можно сформировать требования к функциональности разрабатываемой системы автоматической проверки заданий в системах дистанционного обучения.

Возможности преподавателя:

- авторизация;
- добавления нового курса (дисциплины);
- добавления нового задания;
- просмотр полной статистики по студентам.

Возможности студента:

- регистрация;
- авторизация;
- выбор курса (дисциплины);
- выбор задания для решения;
- отправка решения (программы) на проверку;
- просмотр статистики, результатов тестирования предложенного решения.

Автоматическая система должна проверять следующие типы заданий:

- задачи по программированию;
- документы, представленные в формате приложений MS Office.

Также система должна удовлетворять следующим требованиям:

- Надежность — задача студента должна быть проверена, с занесение

соответствующих результатов в базу данных. Студент должен иметь возможность посмотреть результаты.

- Удобство администрирования — администратор имеет возможность добавления новых заданий, наборов тестов. Администратор (преподаватель) должен видеть полную статистику по студентам.

- Корректность — система должна одинаково обрабатывать решения всех студентов. Задачи, добавленные в систему для проверки должны обрабатываться в порядке поступления. Данные выводимые в таблицу результатов должны полностью отображать информацию, полученную в ходе тестирования программы, выводить перечень пройденных/не пройденных тестов.

- Автономность — система автоматической проверки заданий должна работать круглосуточно, без вмешательства администратора.

- Автоматическое восстановление после сбоев — при обнаружении сбоев в работе автоматическая система проверки заданий должна самостоятельно их устранять.

Выполнить выдвинутые требования можно двумя основными способами:

1. Приобретение и настройка готовых программных решений.
2. Разработка новой информационной технологии.

Обзор готовых решений:

- Moodle — система дистанционного обучения, позволяет создавать обучающие сайты, содержащие методические материалы. Оценка знаний студента происходит путем тестирования.

- Ejudge, conteste — системы, основанные на олимпиадном программировании, реализуют проведение турниров. Данные системы осуществляют проверку заданий по программированию.

Вывод: систем, которые полностью удовлетворяют нашим требованиям, нет. Следовательно, необходимо разработать новую информационную технологию.

Глава 2 Реализация системы автоматической проверки заданий

2.1 Построение модели новой технологии

2.1.1 Модель «Как должно быть»

Автоматизированная проверка заданий по информатике и программированию, базируется на следующих компонентах:

- задания;
- метод проверки задания;
- автоматическая система управления процессом проверки.

Приведем примеры моделирования системы, используя функциональные языки и языки объектного моделирования.

Декомпозиция процессов в функциональной модели осуществляется по принципу «сверху-вниз», когда функциональные блоки разбиваются на подфункции, таким образом, реализуется модульное проектирование автоматизированной информационной системы.

В контекстной диаграмме IDEF0 проектируемая система представлена совокупностью взаимосвязанных работ или функций, не зависящих от объектов, которыми они оперируют. Методология IDEF0 позволяет четко смоделировать логику и взаимодействие процессов в автоматизированной системе проверки заданий.

Процесс моделирования в контекстной диаграмме IDEF0 начинается с описания системы в целом, что является абстрактным уровнем. Далее осуществляется функциональная декомпозиция контекстной диаграммы, то есть система разбивается на части, каждая из которых является подсистемой и описывается отдельной диаграммой декомпозиции.

Диаграмма проверки заданий, представленная на рисунке 2.1., демонстрирует входные и выходные данные в виде потоков, обрабатываемых в автоматизированной системе проверки заданий. Проверка решений осуществляется с учетом критериев и эталонов решений. При рассмотрении

общих случаев – загрузке программного кода задания и других решений на этапах обработки данных, а также на выходе получаем результаты тестирования.

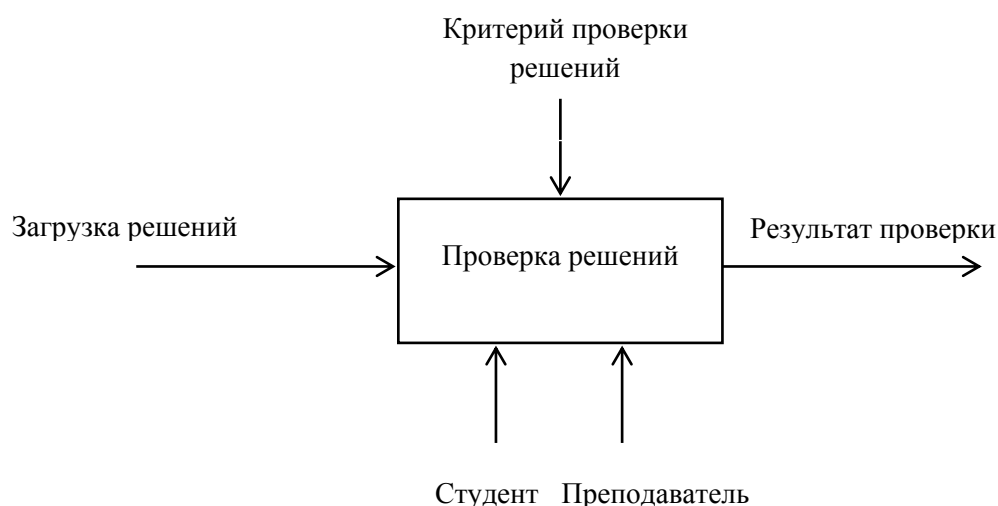


Рисунок 2.1 – Контекстная диаграмма

Диаграммы потоков данных демонстрирует иерархическую структуру функциональных процессов, которые связываются потоками данных. Что отображает процесс преобразования входных данных в выходные, а также показывает отношения между данными процессами. Рассматриваемая модель описывает процесс преобразования информации, начиная от ввода в систему проверки и до вывода пользователю. Внешние сущности в системе порождают информационные потоки данных, потоки переносят информацию к подсистемам, процессам или потребителям информации — внешним сущностям.

Диаграмма потоков данных DFD, представленная на рисунке 2.2, показывает организацию добавления решений заданий, и их сравнения с эталонными результатами в автоматизированной системе. На ней отображены две сущности: Преподаватель и Студент, который загружает своё решение задания в автоматизированную систему, далее информация проходит обработку и добавляется в базу данных. Информация о заданиях и контрольных решениях

добавляется преподавателем, на основании чего производится сопоставление решений Студента с контрольными образцами.

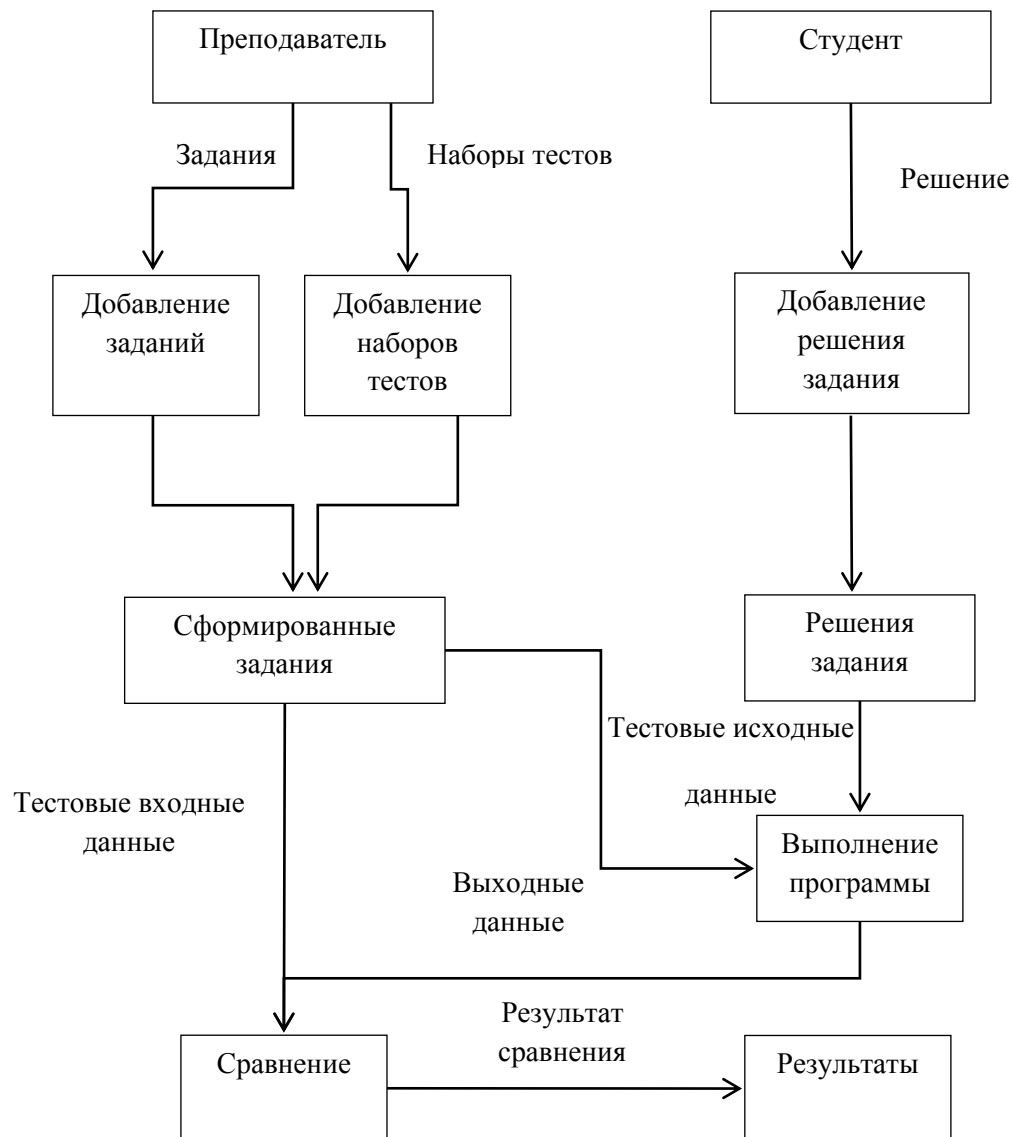


Рисунок 2.2 – Диаграмма декомпозиции DFD

Объектная модель на языке UML строится после рассмотрения функциональной модели автоматизированной системы. Визуальное моделирование UML демонстрирует процесс декомпозиции от общей концептуальной модели к логической, а потом и к физической модели системы.

Диаграмма вариантов использования в UML, представленная на рисунке 2.3, отображает отношения между преподавателем и студентом и позволяет обрисовать систему на концептуальном уровне. Часть дублирующийся

информации в диаграмме вариантов использования устраняется с внедрением взаимосвязей: включение «include» и расширение «extend».

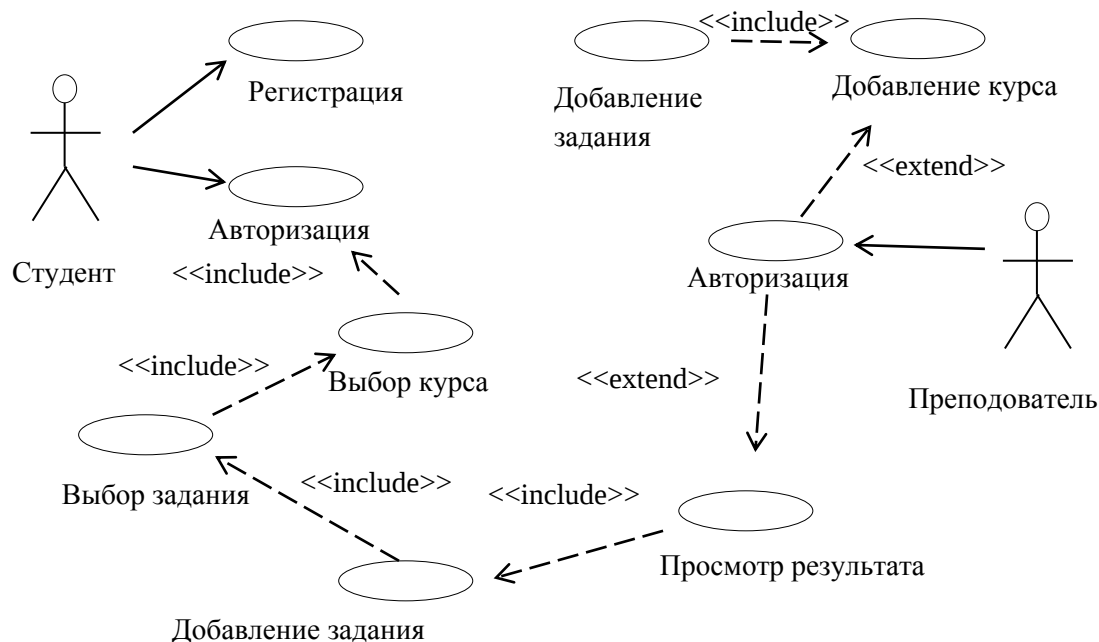


Рисунок 2.3 – Диаграмма вариантов использования UML

Далее рассматривается частный случай системы, которая предоставляет задания по программированию и реализует их проверку и вывод результатов тестирования.

2.2 Проектирование базы данных

2.2.1 Концептуальное проектирование

Концептуальная модель отражает предметную область, для которой проектируется база данных. Для единообразия разработки баз данных сформулированы следующие понятия в концептуальном проектировании.

1. Объект или сущность принято обозначать прямоугольником.
2. Атрибут характеризует объект, отвечает за информацию, которую необходимо хранить об объекте, атрибут принято обозначать овалом.
3. Для обозначения связи или отношения между объектами, используется ромб.

На рисунке 2.4 представлена концептуальная модель для системы автоматической проверки заданий. В модели представлены объекты:

- группа;
- студент;
- решение;
- задача;
- курс.

Объекты, представленные в концептуальной модели, имеют между собой связь «один ко многим». Каждый объект разрабатываемой системы имеет набор атрибутов.

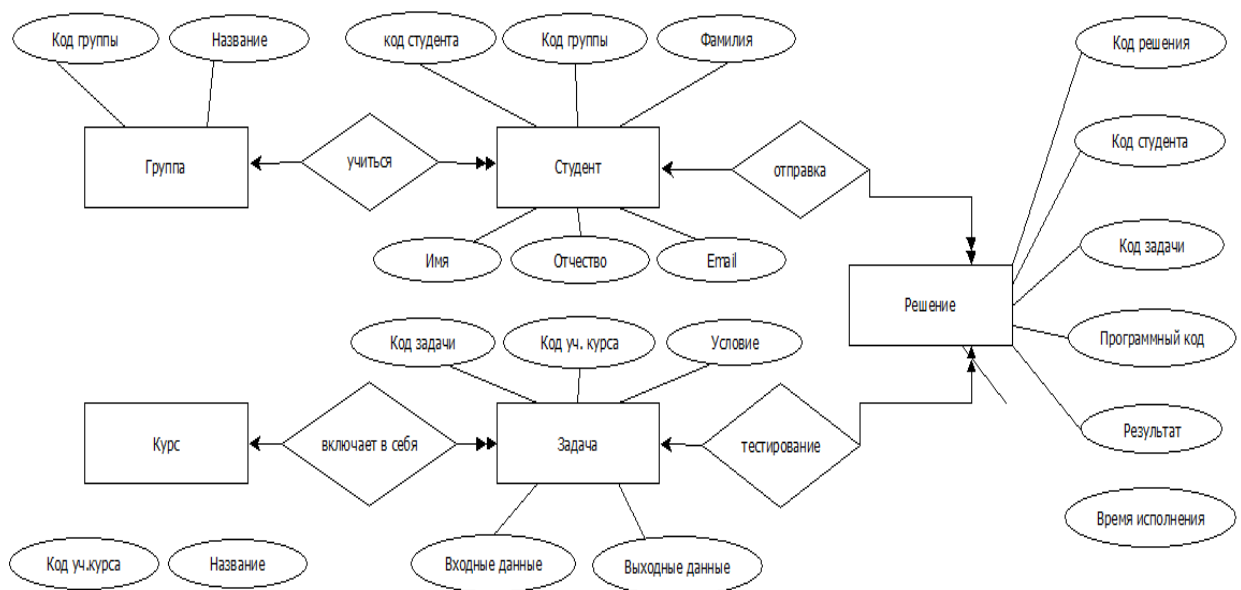


Рисунок 2.4 – Концептуальная модель базы данных

2.2.2 Логическая модель базы данных

Логическая модель – графическое представление структуры базы данных с учетом принимаемой модели данных (иерархической, сетевой, реляционной и т.д.), независимое от конечной реализации базы данных и аппаратной платформы. Иными словами, логическая модель показывает, что

хранится в базе данных (объекты предметной области, их атрибуты и связи между ними), но не отвечает на вопрос как.

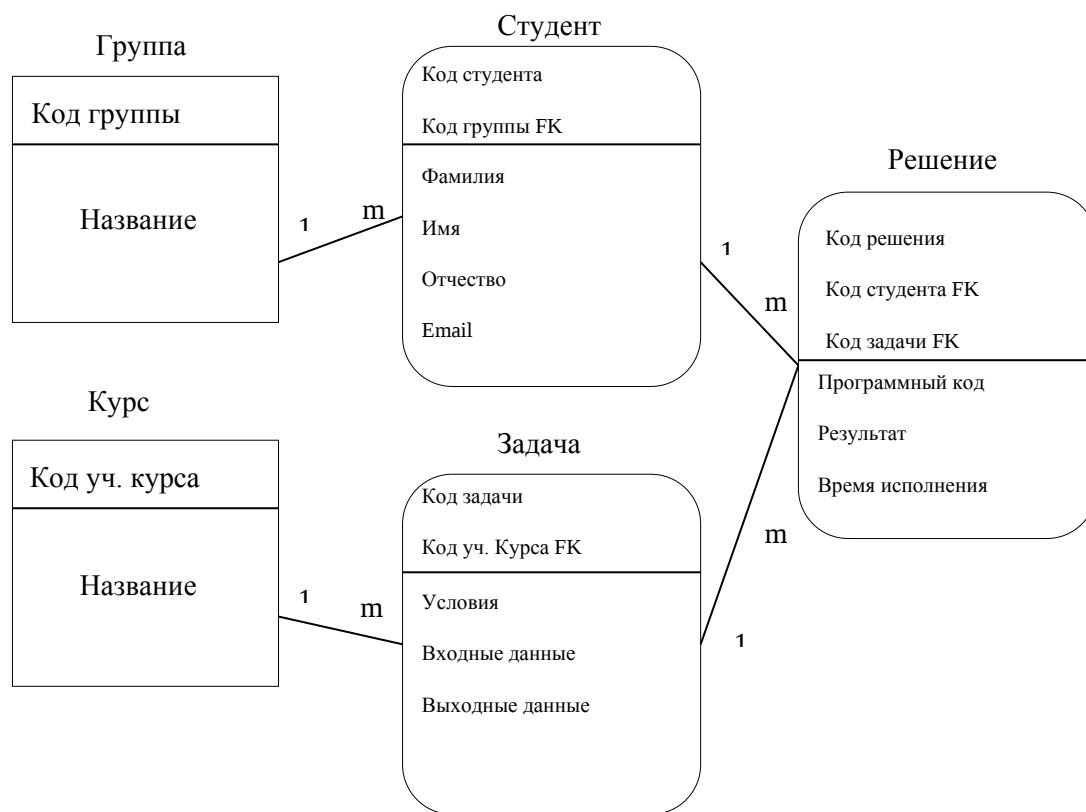


Рисунок 2.5 – Логическая модель базы данных

На рисунке 2.5 представлена логическая схема базы данных для разрабатываемой системы автоматической проверки заданий. На данном рисунке представлены сущности: Группа, Студент, Решение, Задача, Курс, данные сущности образуют связи «один ко многим» и объединяются между собой с помощью внешних ключей.

Первичные ключи, используемые в модели:

- код группы (сущ. Группа);
- код студента (сущ. Студент);
- код решения (сущ. Решение);
- код задачи (сущ. Задача);
- код курса (сущ. Курс).

Внешние ключи, используемые в модели:

- код группы (сущ. Студент);
- код студента (сущ. Решение);
- код задачи (сущ. Решение);
- код курса (сущ. Задача).

Основой реляционной модели является понятие нормализованного отношения, то есть таблицы. Таблицы базы данных содержат сущности предметной области и их свойства:

- нет одинаковых строк, они различаются по первичному ключу (уникальному идентификатору);
- строки-записи не упорядочены сверху вниз;
- столбцы не упорядочены слева направо;
- строки и столбцы при операциях с таблицей просматриваются в произвольной последовательности.

Таблица содержит несколько ключей: первичный ключ, с помощью которого осуществляется связь, и альтернативный. Свойства ключа:

- уникальность (в таблице отсутствуют строки с одинаковыми ключами);
- не избыточность (свойства уникальности пропадает из ключа при удалении его атрибута).

Реляционная база данных показывает множество связанных отношений, с помощью вторичных ключей (Foreign key), т.е. атрибутов, являющихся в других отношениях первичными ключами (Primary key). Ограничения целостности реляционной модели следующие:

- целостность объектов – набор атрибутов первичного ключа принимает определенное значение;
- вторичные ключи принимать значения, которых есть среди значений первичных ключей в связанной таблице.

2.3 Физическое проектирование базы данных

2.3.1 Обоснование СУБД

Система управления базами данных (СУБД) — совокупность программных средств реализующих создание и управление использованием баз данных.

Для разработки системы требуется СУБД выполняющая ряд требований:

- потребление минимального количества ресурсов;
- легкость переноса;
- не коммерческая;
- поддержка требуемых функций.

MySQL считается свободной, относительно небольшой и быстрой реляционной системой управления базами данных. Данная СУБД портирована на большое количество современных платформ и имеет API для широкого спектра языков программирования.

Основные преимущества MySQL:

- обработка нескольких запросов одновременно;
- ODBC драйвер;
- реализована гибкая система паролей и привилегий;
- масштабируемость;
- быстрая работа;
- присутствует утилита проверки и ремонта таблиц;
- все поля в таблицах имеют заданные по умолчанию значения;
- реализована функция резервного копирования;
- обеспеченна хорошая поддержка на стороне хостинг центров.

2.3.2 Физическая модель базы данных

Логическая модель базы данных, которая выражается в терминах языка описания данных СУБД, является физической моделью базы данных.

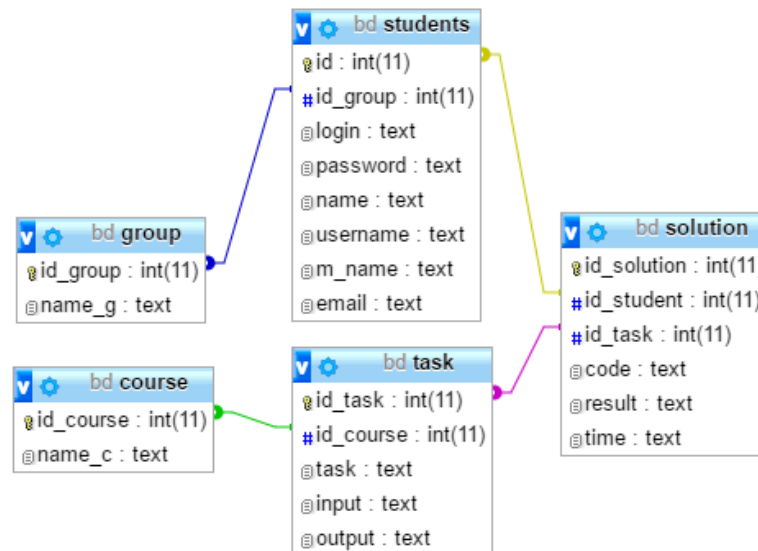


Рисунок 2.6 – Физическая модель базы данных

Все необходимое для создания базы в конкретной СУБД отображает физическая модель базы данных, представленная на рисунке 2.6:

- названия столбцов;
- названия таблиц;
- формат типов данных;
- определение первичных ключей;
- определение внешних ключей.

Основой построения физической модели является логическая на основе ограничений, которые накладываются выбранной СУБД.

Во-первых, имена сущностей и полей записываются латинскими буквами, начиная с буквы и без пробелов.

Во-вторых, у каждого атрибута определяется тип данных, в котором представлены его значения.

В-третьих, в реляционной модели данных требуется преобразовать связи «многие ко многим» и иерархию наследования, так как их использование допускается на уровне логической модели.

2.4 Реализация серверной части

Исходя из сформированных требований к системе автоматической проверки заданий, необходимо создать клиент-серверное приложение, на стороне сервера должны осуществляться следующие функции:

- поддержка одновременного подключения студентов;
- автоматическая проверка предоставленного решения и вывод результата студенту;
- работа с базой данных, подключение, отправка запросов и получение ответов от БД.

Физическая архитектура системы состоит из четырех узлов. Основу составляет классическая трехзвенная архитектура:

- тонкий клиент;
- web-сервер;
- СУБД.

Дополнительным узлом в системе автоматической проверки заданий является проверка присланных решений. Преподаватель и студент осуществляют работу с информационной системой через web-браузер. Все программы отправленные студентами на проверку, сохраняются в базе данных.

Рассмотрим подробнее проектирование и реализацию механизма автоматической проверки решений присланных студентом. Студент через форму на странице ИС отправляет решение задачи на сервер, для дальнейшей проверки. На сервере скрипт тестирования формирует запрос к базе данных на получение набора тестов для проверяемой задачи. Скрипт получает ответ из базы данных, после чего запускает процесс тестирования (проверки) задания. Итогом проведения проверки, является генерируемый результат тестирования. Далее результат тестирования записывается в базу данных. Студент может посмотреть результат тестирования программы на странице ИС.

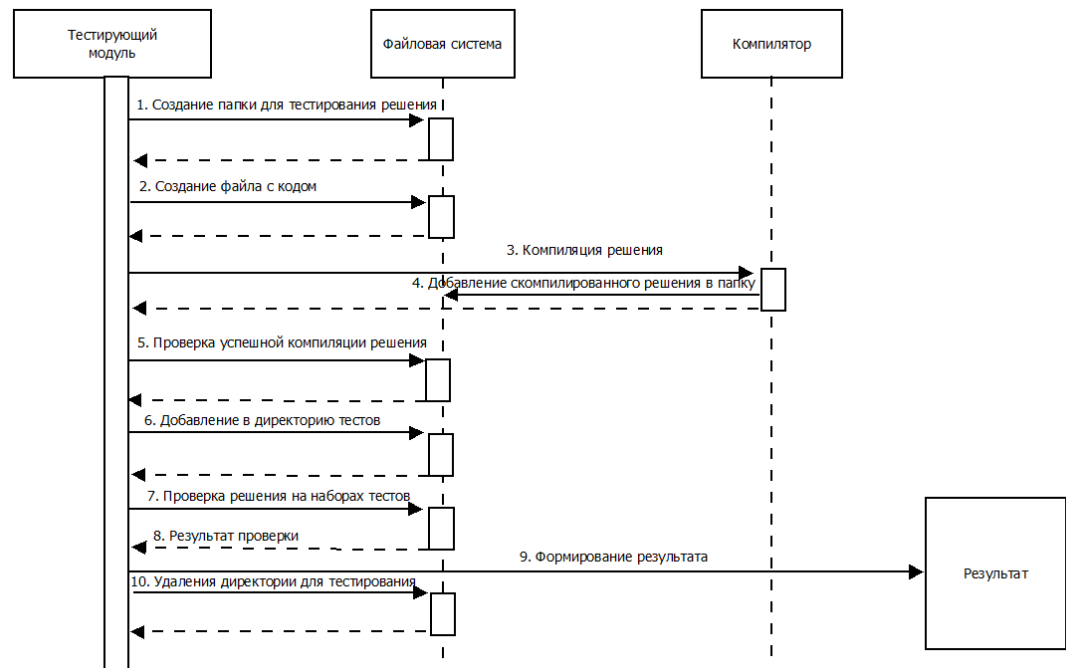


Рисунок 2.7 – Диаграмма последовательности для механизма тестирования задачи проверяющей системой

На рисунке 2.7 процесс тестирования показан более детально, а рисунок 2.8 показывает реализацию проверки решения. После запуска проверки решений в заранее определенной директории создается папка для тестирования. Имя созданной папки является уникальным, т. к. оно эквивалентно ID решения. Далее система формирует файл, содержащий код решения, данный файл необходимо отправить на компиляцию, и, в случае успешной компиляции, полученный файл необходимо сохранить в папке тестирования. Если процесс компиляции закончился неудачей, то студент получает соответствующие сообщение, иначе тестирование продолжается. В созданную папку добавляются наборы тестов, проверка решения производится последовательно на всех наборах тестов. Если в результате прохождения какого-то теста решение не получает результат «Принято», то проверка останавливается и студенту, на странице ИС содержащей информацию по тестированию решения, выводится соответствующие сообщение о результатах тестирования. Если решение получает результат «Принято» на всех наборах тестов, то студенту выводится сообщение об успешной сдаче решения. После окончания процесса

тестирования решения, созданная ранее директория удаляется. Все шаги процесса тестирования производятся проверяющей системой в автоматическом режиме.

```
<?php
session_start();
$student_id = $_POST['student_id'];
$task_id = $_POST['task_id'];
$input_path= '../files/'.$task_id.'/right/input.txt';
$output_path= '../files/'.$task_id.'/right/output.txt';
mkdir('../files/'.$task_id.'/'.$student_id, 0777, true);
if(file_exists('../files/'.$task_id.'/'.$student_id.'/output.txt')) {('ERROR!');}
else{
    fclose(fopen('../files/'.$task_id.'/'.$student_id.'/output.txt','x'));
}
$allowed_filetypes = array('.java','.c');
$max_filesize = 524288;
$upload_path = '../files/'.$task_id.'/'.$student_id.'/';
$filename = $_FILES['qwe']['name'];
$ext = substr($filename, strpos($filename, '.'), strlen($filename) - 1);
if (!in_array($ext, $allowed_filetypes)) die('Данный тип файла не поддерживается.');
```

```
if (filesize($_FILES['qwe']['tmp_name']) > $max_filesize) die('Файл слишком большой.');
```

```
if (!is_writable($upload_path)) die('Невозможно загрузить файл в папку.');
```

```
if (move_uploaded_file($_FILES['qwe']['tmp_name'], $upload_path . $filename))
    echo 'Ваш файл успешно загружен <a href="'. $upload_path . $filename . '">Подробнее</a>';
else echo 'При загрузке возникли ошибки. Попробуйте ещё раз.';
$pos = strpos($filename, ".");
$fn = substr($filename, 0, $pos);
copy($input_path, $upload_path.'input.txt');
```

```
exec('C:\MinGW\bin\gcc '. $filename .' -o '.$fn);
exec($fn);
if(file_exists('../files/'.$task_id.'/'.$student_id.'/'.$filename)){
    $output_right = file_get_contents('../files/'.$task_id.'/right/output.txt');
    $output_student = file_get_contents('../files/'.$task_id.'/'.$student_id.'/output.txt');
```

```
similar_text($output_right, $output_student, $percent);
if(isset($percent)){
    include '../db.php';
    if($percent == 100){
        $answer = 1;
    }
    else{
        $answer = 0 ;
    }
    $sql = "INSERT INTO results ". "(id_task,id_student,percentage) ". "VALUES('$task_id','$student_id','$answer)";
    $retval = mysql_query($sql, $link);
}
```

Рисунок 2.8 – Реализация проверки решения

Вывод: данная глава раскрывает этапы построения модели новой технологии, с учетом предъявляемых к ней требований, то есть построение модели «Как должно быть».

Глава 3 Развертывание и тестирование системы автоматической проверки заданий

3.1 Тестирование разработанной системы и описание процессов

На рисунке 3.1 представлена главная страница ИС, которая имеет следующие разделы:

- приветствие и переход на вкладку с регистрацией;
- форма для авторизации студента;
- ссылка на методические материалы;
- ссылка, содержащая примеры с оформлением решений;
- ссылка на инструкцию по работе с системой;
- ссылку на контакты.

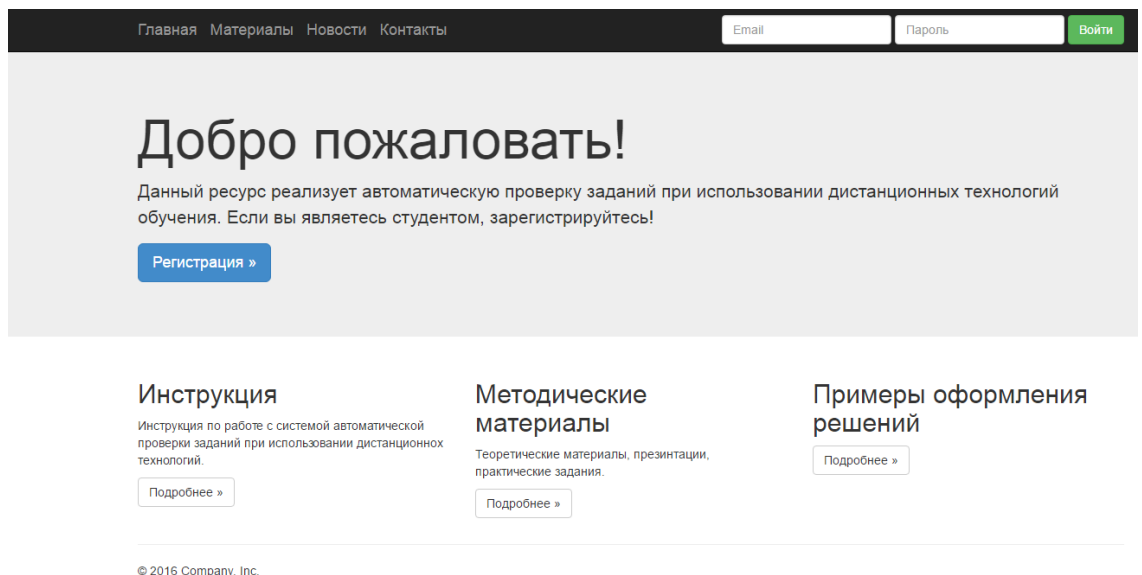


Рисунок 3.1 – Главная страница ИС

При переходе на вкладку с регистрацией рисунок 3.2, пользователю необходимо заполнить поля.

Пожалуйста зарегистрируйтесь

Логин

Пароль

Группа

Имя

Фамилия

Отчество

Email

Регистрация

Рисунок 3.2 – Регистрационная форма пользователя

После ввода всех данных пользователю придет письмо с сообщением об успешной регистрации. А данные введенные пользователем добавятся в базу данных как показано на рисунке 3.3.

id	id_group	login	password	name	username	m_name	email
1	1	gena	123456	Геннадий	Юрковец	Юрьевич	urkovecgennadii@yandex.ru

Рисунок 3.3 – Запись пользователя в БД

У зарегистрированного пользователя есть возможность авторизоваться в ИС, после авторизации студенту становится доступен вход в личный кабинет (рисунок 3.4). Где ему необходимо выбрать курс, на котором он учиться и изучаемую дисциплину. Выполнив данные пункты, студент получает информацию о своей успеваемости. Таблица представляет собой перечень задач с их коротким описанием. Студент может отослать свое решение задачи на проверку, для этого нужно выбрать задачу и пройти по ссылке во вкладке «Выполнить». Студент имеет возможность посмотреть результаты тестирования, нажав на кнопку «Подробнее» во вкладке информация. Также

каждая задача имеет статус выполнения для наглядности.

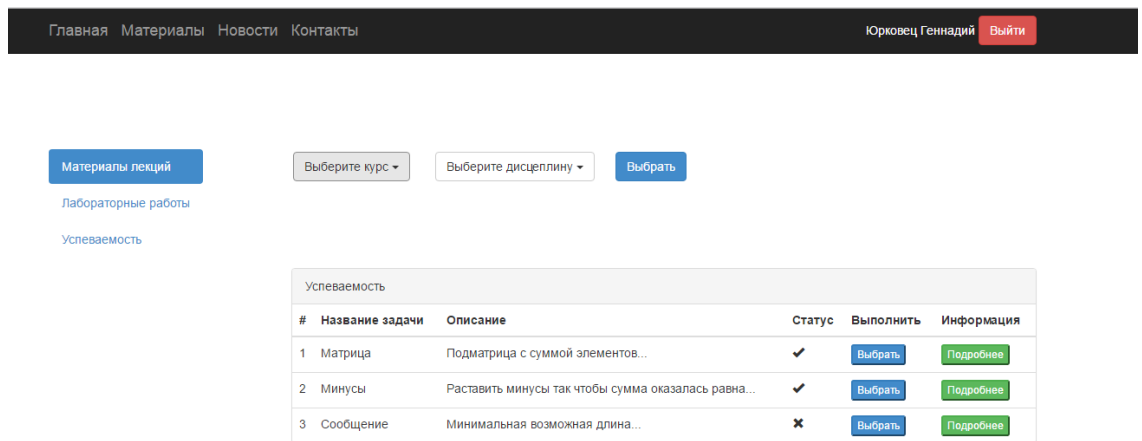


Рисунок 3.4 – Личный кабинет студента

Следующей представлена страница с конкретной задачей и формой для отправки студентом его решения. Пункты, содержащиеся на странице (рисунок 3.5):

- номер и название задачи;
- условие задачи;
- ФИО студента;
- форма для прикрепления файла;
- форма для выбора языка программирования;
- ссылки для переходов на страницы с информацией (оформление решения, информация о результатах проверки решения);
- кнопка для отправки формы.

Требуется сложить два целых числа A и B.

Входные данные

В единственной строке входного файла INPUT.TXT записано два натуральных числа через пробел, не превышающих 10^9 .

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно целое число — сумму чисел A и B.

Пример

№	INPUT.TXT	OUTPUT.TXT
1	2 3	5

Отправить решение

Студент	Юрковец Геннадий Юрьевич	
Файл задачи	 Обзор...	Как оформлять код?
Язык программирования	Выбрать	Что означают результаты проверки решений?
Отправить		

Рисунок 3.5 – Страница с конкретной задачей и формой для отправки решения

В качестве файла решения студент отправляет файл с именем Main.java. Решение задачи, написанное студентом на Java должно содержать только один public класс с именем Main.

Пример кода содержавшегося в файле правильного решения задачи "A+B" на Java (рисунок 3.6).

```
import java.util.*;
import java.io.*;

public class Main{
    public static void main(String[] argv) throws IOException{
        new Main().run();
    }
    PrintWriter pw;
    Scanner sc;
    public void run() throws IOException{
        sc = new Scanner(new File("input.txt"));
        int a=sc.nextInt(), b=sc.nextInt();
        pw = new PrintWriter(new File("output.txt"));
        pw.print(a+b);
        pw.close();
    }
}
```

Рисунок 3.6 – Пример кода задачи

Для обработки задачи, присланной студентом на проверку, системе требуется некоторое время. Студент может посмотреть результат тестирования задачи, через некоторое время. Для просмотра результата необходимо перейти по вкладке «Подробнее», которая представлена на рисунке 3.7.

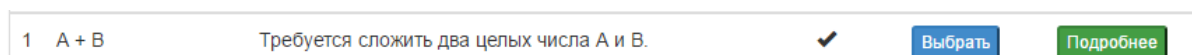


Рисунок 3.7 – Вкладка «Подробнее»

Результат тестирования правильно написанной программы, на рисунке 3.8 в таблице представлены этапы тестирования:

1. Программа успешно скомпилировалась.
2. Программа прошла все наборы тестов корректно.
3. Выполнение программы укладывается в заданные временные рамки.

Требуется сложить два целых числа A и B.

Входные данные

В единственной строке входного файла INPUT.TXT записано два натуральных числа через пробел, не превышающих 10^9 .

Выходные данные

В единственную строку выходного файла OUTPUT.TXT нужно вывести одно целое число — сумму чисел A и B.

Результат тестирования

№	Результат	Статус
1	Ошибка компиляции	✗
2	Неверный ответ	✗
3	Неуложилась по времени	✗

Программа успешно прошла тесты примеров из условия задачи.

Рисунок 3.8 – Результат тестирования задачи

Администратор имеет возможность просмотра общей статистики по студентам. Для просмотра статистики ему необходимо выбрать курс и группу. Статистика представлена в виде таблицы на рисунке 3.9, у всех студентов

отображается статус по каждой задаче и общее количество выполненных работ.

Главная Материалы Новости Контакты Преподаватель Выйти									
Материалы лекций Лабораторные работы Успеваемость Выберите курс ▾ Выберите группу ▾ Выбрать									
Успеваемость									
#	ФИО	Номер задания	1	2	3	4	5	Итог	Информация
1	Юрковец Геннадий Юрьевич		✓	✗	✓	✗		2	Подробнее
2	Иванов Иван Иванович		✓	✗	✓	✓		3	Подробнее
3	Пертов Петр Петрович		✓	✓	✓	✓		4	Подробнее

Рисунок 3.9– Страница общей статистики по студентам

3.2 Развертывание системы

Для развертывания системы автоматической проверки заданий необходимо выполнить следующие пункты.

1. Установить windows server.
2. Установить и настроить связку связку Apache + PHP + MySQL + phpMyAdmin.
 - первым делом необходимо скачать дистрибутив Apache с сайта;
 - после того как дистрибутив Apache закончит загрузку, необходимо открыть архив и распаковать его в папку;
 - редактирование конфиг файла;
 - установка Apache используя командную строку;
 - открыть директорию bin и запустить файл ApacheMonitor.exe. В системном трее появится значок Apache, который позволяет запускать и останавливать службу Apache, нажать кнопку «start» для запуска службы;
 - проверка работоспособности;
 - открыть страницу загрузки дистрибутива MySQL;
 - скачать дистрибутив, запустить установку службы;
 - пройти процесс установки службы, проверить работоспособность;

- открыть страницу загрузки phpMyAdmin;
 - выбор архива для скачивания;
 - создать папку `phpmyadmin` в директории `Apache\htdocs` и извлечь файлы скаченного архива;
 - необходимо создать конфигурационный файл для MySQL.
3. Установить компилятор MinGW.
 4. Создать папку для системы проверки заданий на сервере, добавить в папку файлы системы.
 5. Создать базу данных.
 6. Создать таблицы в базе данных, используя SQL запросы (Приложение Б).

Вывод: данная глава описывает этапы тестирования разработанной системы автоматической проверки заданий, с учетом предъявляемых к ней требований, также описан процесс взаимодействия студента и преподавателя с системой, описан контрольный пример работы системы и процесс развертывания.

Заключение

Выполненная работа посвящена исследованию использования дистанционного обучения (ДО) и проблем контроля заданий в системах дистанционного обучения, проверки практических заданий. Цель работы определила ее основное направление – автоматизация проверки заданий при использовании дистанционных технологий. В результате проведенного исследования описаны положительные стороны ДО и выявлены основные проблемы, разработаны концептуальная, логическая и физическая модель базы данных для системы автоматической проверки заданий.

В ходе выполнения работы были решены следующие теоретические задачи:

1. Проведено исследование проблемы автоматической проверки заданий при дистанционном обучении, в ходе которого выявлены основные характеристики и определены пути решения выявленных проблем.

2. Изучена общая информация по реализации дистанционного обучения в образовательных учреждениях, и определен спектр взаимодействия обучающегося и преподавателя.

3. Рассмотрены этапы обучения, существующий процесс отправки заданий студентом и проверки заданий преподавателем.

В процессе выполнения проектных задач бакалаврской работы:

1. Построена модель новой технологии, обладающая новыми требованиями и особенностями.

2. Разработана система автоматической проверки заданий на разных тестах.

3. Выполнено тестирование разработанной системы и описан процесс ее развертывания.

Список используемой литературы

Учебники и учебные пособия

1. Фримен Э. Паттерны проектирования. / Э. Фримен -СПб.: Питер, 2011. - 645 с.
2. Виштак Н.М. Информационное обеспечение научно-исследовательской работы студентов. Т. 5. / Н.М. Виштак - М: Научные труды SWorld, 2013. - 620 с.
3. Виштак О.В. Направления программной реализации электронных образовательных ресурсов. Т. 2. / О.В. Виштак - М: Научные труды SWorld, 2013. - 540 с.
4. DuBois, P. MySQL – Developer's Library / P. DuBois. - Addison-Wesley Professional, 2013. - 340 с.
5. Kaplan Andreas, Haenlein Michael. Higher education and the digital revolution: About MOOCs, SPOCs, social media, and the Cokie Monster/ Andreas Kaplan, Michael Haenlein, 2016. - 325 с.
6. Galusha Jill M. Barriers to Learning in Distance Education. / Galusha Jill M., 2016. - 425 с.
7. Schwartz B. Hight Perfomance MySQL: Optimization, Backups and Replication / B. Schwartz, P. Zaitsev. – O'Reilly Media, 2012. – 360 с.

Статьи

8. Аверченко Л. К. Дистанционная педагогика в обучении взрослых / Л. К. Аверченко // Философия образования. - 2011. - № 6 (39). - С. 322-329.
9. Аксютин П.А. Опыт построения среды электронного обучения и ее использование для преподавания дисциплины «Информационные технологии» / П.А. Аксютин // Электронное обучение в вузе и в школе: Материалы сетевой международной научно-практической конференции. – СПб.: Астерион - 2014. – С. 28–31.
10. Верещагин А.Г. Автоматизация тестирования программ как средство повышения эффективности учебного процесса. / А.Г. Верещагин

//Известия МГИУ – 2012 - № 3 (27), с. 28–41.

11. Виштак Н.М. Функционально-структурная модель интеллектуальной обучающей системы / Н.М. Виштак, Д.А. Фролов, Е.В. Варгина // Научный журнал «Фундаментальные исследования». М.: ИД «Академия естествознания» - 2013. — С. 871—874.

12. Гриневич Е. А. Методика дистанционного изучения информатики студентами экономических специальностей/ Е. А. Гриневич //Информатизация образования. - 2011. - №1. - С. 36-44.

13. Козлов Е.А. «Учебный процесс в системе дистанционного образования и особенности методик контроля знаний при дистанционном обучении» / Е.А. Козлов // Вестн. Вол. Ун-та. Им. Татищева. – 2009. –№ 13. – С. 2—4. Режим доступа: <http://cyberleninka.ru/article/n/uchebnyy-protsess-v-sisteme-distantionnogo-obrazovaniya-i-osobennosti-metodik-kontrolya-znaniy-pri-distantionnom-obuchenii#ixzz4Bfycb4f4>

14. Станкевич А.С. Общий подход к проведению итогов соревнований по программированию при использовании различных систем оценки. / А.С. Станкевич //Компьютерные инструменты в образовании - 2011 - № 2. - С. 27–38.

15. Ольнев А. С. Использование новых технологий в дистанционном обучении / А. С. Ольнев // Актуальные проблемы современной науки. - 2011. - № 1. - С. 96.

16. Дячкин О. Д. Опыт разработки методики компьютерного обучения математике/ О. Д. Дячкин // Открытое и дистанционное образование. - 2009. - № 4. - С. 24-30.

17. Топунова М. К. Оценка качества знаний учащихся при дистанционном обучении в системе начального общего образования / М. К. Топунова // Дистанционное и виртуальное обучение. - 2012. - № 2. - С. 31-42.

18. Филинов А.Н. Система автоматического тестирования Ejudge. / А.Н. Филинов //Информатика и образование – 2012 - № 9 - С. 63–64.

19. Чошанов М. А. Обучающие системы дистанционного образования /

Чошанов М. А. // Школьные технологии. - 2011. - № 4. - С. 81-88.

20. Anderson T. Three generations of distance education pedagogy. / Terry Anderson, Jon Dron //International Review of Research on Distance and Open Learning – 2011 - №12(3) – С.12-13.

Электронный ресурс

21. Андреева Е.В., Принципы проверки учебных и олимпиадных задач по информатике. [Электронный ресурс] : учебное пособие / Е.В. Андреева – Электрон. Дан. - М. - Режим доступа : <http://inf.1september.ru/2001/9/art/35.html>

22. Система для проведения турниров и индивидуального решения задач по олимпиадному программированию Contester. URL: <http://www.contester.ru> (дата обращения 1.06.2016).

23. Швецов А. Н Отчет (Введение, разделы 1,2,3). [Электронный ресурс] : учеб. пособие. - М. - Электрон. дан. – Режим доступа : <http://int.vstu.edu.ru/files/etapshvecov/otchet4.doc>.

Приложение А

Перечень тестов для дальнейшей проверки программ

Рассмотрим описание тестовых данных и самого процесса тестирования для оценки правильности работы и качества программ студентов.

1. Первым тестом происходит проверка работоспособности программы, поэтому тест можно сделать максимально простым. В качестве первого теста можно использовать тест, взятый из условия задачи. Данный вид теста помогает преподавателю определить понимание студентом условия задачи, соответствие условию входных и выходных данных в программе, правильное указание входного и выходного файла (если условие предполагает ввод/вывод данных через файл). Если программа не проходит данный вид теста, то студенту будет предоставлена информация об ошибке: `FormatError` – нарушение формата представления выходных данных.

2. Вторая группа тестов направлена на отслеживание так называемых вырожденных случаев. Данные случаи возникают, когда решение задачи или не существует (в условии задачи заранее обговаривается сообщение выводимое программой в данной ситуации), или полностью отличается от предложенного решения, опиравшегося на основной алгоритм. Данная группа в первую очередь содержит нулевые примеры. При работе с числовыми входными данными это обычно нулевые значения (если условие задачи не противоречит этому), в случае текстовых входных данных, нулевым примером является использование пустых входных файлов, а также использование последовательностей из пробелов и символов переноса строки.

Также примером вырожденности является нарушение совокупности входных данных, при которой необходимо специально обрабатывать данные.

Если в условии задачи необходимо решить уравнение $ax^2 + bx + c$, где a, b, c , принадлежат множеству действительных значений, то при условии, где, $a=0$ квадратное уравнение приобретает линейный вид. Программа студента должна учитывать данный случай.

Тесты на «вырожденность», необходимые для проверки данной задачи:

$a = 0, b = 0, c = 0;$

$a = 0, b = 0, c = 1;$

$a = 0, b = 1, c = 2 .$

Координаты N точек на плоскости являются входными параметрами. Найти точку на данной плоскости, расстояние от которой до наиболее удаленной из N точек минимально. В ходе решения данной задачи случаи $N = 1$ и $N = 2$ рассматриваются независимо друг от друга. Входные данные должны обрабатываться корректно, иначе возможны следующие виды ошибок при автоматическом тестировании.

`RuntimeError` — ошибка выполнения программы (возникает в результате деления на ноль).

`WrongAnswer` — неверный ответ (студент не учитывает, что при вводе вырожденных входных данных предложенный им алгоритм обработки имеет отличия от основного алгоритма).

3. Данная группа тестов организывает проверку граничных случаев (входные и выходные данные принимают граничные значения). Задача этой группы тестов — выявить возможные программные ошибки, которые могли быть допущены студентом при реализации, даже при использовании правильного алгоритма.

В самом начале, следует проверять, что в ходе написания программы студент не допустил ошибки при выборе подходящих типов данных для переменных. Так, если программа должна вычислять сумму двух целых чисел, числа по модулю не больше 32 767, то данные числа могут представляться, используя двухбайтовый знаковый тип данных, однако для полученного результата этот тип данных не подходит.

При решении задач с использованием численных методов, к граничным относятся те тестовые данные, в ходе обработки которых необходима максимальная точность при вычислении. К примеру, большие или очень маленькие, но не равные нулю значения, используемые для входных

параметров. Часто критичным фактором для работоспособности является количество оперативной памяти, которое задействовано во время выполнения программы. Если располагать все переменные в статической памяти, то появится часто допускаемая при программировании ошибка — превышение границы массива при выполнении программы, данная ошибка может появиться в случае использования максимального объема входных данных. Если в условии прописано, что при использовании максимального допустимого количества данных студент должен использовать возможности динамической памяти, необходимо продумать и написать набор тестов для проверки программы, работающей по данному сценарию. В ходе работы программы студент может неверно выделить оперативную память или неправильно к ней обратиться. Также студент может неэффективно использовать память в своей программе. В этом случае, легко, подобрать тест, используя который можно выявить, что программа студента работать не будет в силу недостатка динамической памяти, при этом эталонная программа в тех же условиях работает правильно. Программа, работающая с текстом часто не правильно обрабатывает строки, написанные корректно с точки зрения условия, если их длина более 255 символов. Это часто зависит от особенностей применяемого языка программирования или в результате использования массива символов фиксированной размерности, вместо одной или двух переменных символьного типа. Для многих программ, где в ходе решения, где требуется обработать файл, для которого не указано конечное количество строк и чисел, является критичным наличие в конце файла пробелов и пустых строчек. При подготовке тестов необходимо учитывать нахождение признака конца файла за последним символом данных подаваемых на вход, причем данный признак должен быть в той же самой строке.

4. Эта группа тестов направлена на проверку правильности алгоритма решения задачи в целом. Поделить их можно на тесты общей направленности и специальные тесты. Последние тесты направлены на выявление работоспособности программы в том случае, если входные данные будут

организованны специальным образом. Например, набор входных данных может быть отсортирован сначала по возрастанию, а затем по убыванию, хотя по условию данные не предполагают расположение в определенном порядке, однако алгоритм может использовать сортировку входных данных. Или можно сделать равными между собой значения всех параметров. Такие тесты помимо выявления специфики работы программы студента, помогают преподавателю проверить правильность данных подаваемых на вход вручную. Это значительно сократит объём работы связанный с проверкой компьютерных результатов. Тесты общей направленности должны выявить логическую схему алгоритма, все его ветви, такая проверка называется испытанием ветвей.

Если задача использует решение, реализованное с помощью алгоритма на графах, то программу нужно проверять на наборе данных которые соответствуют:

- связанному и не связанному графу;
- графу без циклов;
- графу, все вершины которого принадлежат тому или иному циклу;
- графу с пустым множеством ребер (если это не противоречит условию);
- полному графу и т.д.

Если особенности входных данных не влияют на алгоритм решения задачи, в алгоритме нет логического ветвления, можно использовать один-два общих теста. Например, для решения задания с графами нужно найти степень каждой вершины, то вполне достаточно написать два теста, один из тестов соответствует Эйлерову графу, второй - нет.

Если программа или её части подразумевает разделение и получение разных исходов, то требуется написать тест на каждый из возможных вариантов работы.

5. Если в процессе решения задачи можно прийти к ошибочному применению эвристических алгоритмов, то для тестов необходимо составить

такие наборы данных, в результате использования которых такие приближенные решения будут работать неверно. Часто для проверки задачи приходится составить и реализовать несколько разных эвристических подходов, для проверки каждого из них приходится реализовать отдельный тест. Например, в процессе решения задачи по поиску в графе гамильтонова цикла на небольшом тесте будет видно, что при использовании «жадного» алгоритма решение данной задачи часто будет неправильным. Еще сложнее составить тестовый пример для случая решения задачи с использованием «перебора с предпочтением», когда алгоритм сам заканчивает работу, как только время, отведенное на решение приближается к максимально допустимому. Это означает, что программа реализована с отсечением по времени. При правильной реализации данного перебора, программа зачастую быстро находит правильный вариант решения, а в дальнейшем используя незаконченный перебор, показывает, что предложенное решение является лучшим. Однако с такими проблемами приходится сталкиваться при тестировании программ олимпиадного типа, в которых использование данного приема для решения считается допустимым.

6. После корректной и правильной работы программы на всех выше упомянутых тестах, следует переходить к тестированию эффективности используемых для решения алгоритмов. Данный аспект всегда является самым уязвимым при решении учебных заданий. Зачастую, несмотря на допустимые в условии большие входные данные, алгоритм работает неправильно, причиной этого является упрощение студентом реализации, что влечет за собой неоправданное увеличение вычислительной сложности алгоритма. При этом студент не всегда самостоятельно осознает тот факт, что задача решается некорректно. Если в процессе выбора способа реализации задачи алгоритм полиномиальной сложности был заменен на экспоненциальный, то проверить это не сложно. При реализации алгоритма, вычислительная сложность которого $N!$, где N — количество входных данных, программе не успеет завершить работу за одну минуту даже при условии $N < 16$ (от быстрого действия

компьютера зависит точная верхняя граница работоспособности данного алгоритма, однако ее не возможно повысить более чем на 2, даже при увеличении скорости процессора в 100 раз.). Часто данная ошибка при выборе алгоритма возникает при решении задач по дискретной математике, студент выбирает для решения метод полного перебора вариантов, вместо динамического программирования. В этом случае использование при тестировании средних по объёму входных тестовых данных помогает определить, что программа не укладывается во время, отведенное для решения. При использовании автоматического тестирования это приводит к превышению ограничения по времени. Такая же ошибка, возникающая при прохождении предыдущих тестов означает, заикливание программы.

Если используемый алгоритм остаётся полиномиальным или имеет одинаковый порядок сложности с эталонным алгоритмом решения, и лишь в фиксированное число раз отличается от него, то проверка становится на порядок сложнее. Нужно подобрать входные тестовые данные так, чтобы эталонная программа выполняла работу за половину максимально разрешенного времени указанного в условии. Считается допустимым увеличение студентом вычислительной сложности алгоритма в два раза. Составление таких тестовых наборов входных данных часто организовано специальным образом, так как количество входных данных не всегда влияет на трудоемкость работы программы. Подбор данных для тестирования и проверка работы студента должны быть организованы на идентичных компьютерах.

7. К последней группе относятся случайные тесты. Данные тесты отражают в полном объеме реальные входные данные. Не представляет труда сгенерировать входные данные для этой группы тестов, зачастую достаточно пары тестов. Проверка правильности работы программы на данной группе тестов является сложной. Это связано с трудностью проверки результатов работы программы вручную. Часто приходится реализовывать дополнительную программу. Примером является завершающий тестирование программы, большой случайный тест.

Вывод: глава посвящена вопросам развертывания и тестирования системы автоматизированной проверки заданий, на основе предъявляемых требований.

Приложение Б

Листинг кода

Sql запросы для создания базы данных

```
CREATE TABLE 'bd'.students' ( 'id' INT NOT NULL AUTO_INCREMENT , 'id_group' INT NOT NULL , 'name' TEXT NOT NULL , 'username' TEXT NOT NULL , 'm_name' TEXT NOT NULL , 'email' TEXT NOT NULL , PRIMARY KEY ('id ')) ENGINE = InnoDB;
```

```
CREATE TABLE 'bd'.task' ( 'id_task' INT NOT NULL AUTO_INCREMENT , 'id_course' INT NOT NULL, 'task' TEXT NOT NULL, 'input' TEXT NOT NULL , 'output' TEXT NOT NULL , PRIMARY KEY ('id.task')) ENGINE = InnoDB;
```

```
CREATE TABLE 'bd'.solution' ('id_solution' INT NOT NULL AUTO_INCREMENT , 'id_studen' INT NOT NULL , 'id_task' INT NOT NULL, 'code' TEXT NOT NULL , 'result' TEXT NOT NULL , 'time' TEXT NOT NULL, PRIMARY KEY ('id_solution')) ENGINE = InnoDB;
```

```
CREATE TABLE 'bd'.group' ('id_group' INT NOT NULL AUTO_INCREMENT, 'name_g' TEXT NOT NULL, PRIMARY KEY ('id_group')) ENGINE = InnoDB;
```

```
CREATE TABLE 'bd'.course' ('id.course' INT NOT NULL AUTO_INCREMENT 'name_c' TEXT NOT NULL , PRIMARY KEY (' id_course ')) ENGINE = InnoDB;
```

Соединение с базой данных

```
<?php
$link = mysql_connect('localhost', 'root', '');
$selectd = mysql_select_db("rangedtests", $link)
or die("Could not select DataBase");
```

Содержание файла index.html

```
<?php
session_start();

?>

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <meta http-equiv="X-UA-Compatible" content="IE=edge">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <meta name="description" content="">

  <meta name="author" content="">

  <link rel="icon" href=" ../favicon.ico">

  <title>Главная</title>
```

```

<link href="css/bootstrap.min.css" rel="stylesheet">
<link href="css/ie10-viewport-bug-workaround.css" rel="stylesheet">
<link href="jumbotron.css" rel="stylesheet">
<script src="js/ie-emulation-modes-warning.js"></script>

<script type="text/javascript">
    $(function(){
        $('close').click(function(){
            $("#closebutton").alert('close');
        }
    }
</script>
</head>
<body>
<?php
if(isset($_POST['exit'])){
    session_destroy();
    $_SESSION['isUser']=0;
    $isUser = 0;

}

if(isset($_POST['enter'])){
    include 'db.php';
    $login_web = $_POST['login'];
    $password_web = $_POST['password'];
    $sql = "SELECT * FROM `students` WHERE `login`='$login_web'";
    $retval = mysql_query($sql, $link);
    while ($data = mysql_fetch_array($retval)){
        $id = $data["id"];
        $id_group = $data["id_group"];
        $login = $data['login'];
        $password = $data['password'];
        $name = $data['name'];
        $s_name = $data['s_name'];
    }
}

```

```

$m_name = $data['m_name'];
$email = $data['email'];
}
if($password == $password_web){
    $isUser = 1;
    $_SESSION['isUser'] = 1;
    $_SESSION['id_group'] = $id_group;
    $_SESSION['id'] = $id;
    $_SESSION['login'] = $login;
    $_SESSION['name'] = $name;
    $_SESSION['password'] = $password;
    $_SESSION['s_name'] = $s_name;
}
else{
    $isUser = 0;
    $_SESSION['isUser'] = 0;
    $showpassalert = 1;
}
}
if($showpassalert == 1) {
    ?>
    <div id="closebutton" class="alert alert-danger">
        <span class="close" data-dismiss="alert">×</span>
        Неверный пароль!
    </div>
<?php
}
?>

<nav class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
        <div class="navbar-header">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar" aria-
                expanded="false" aria-controls="navbar">

```

```

</button>

<a class="navbar-brand" href="index.php">Главная</a>

<a class="navbar-brand" href="mater.php">Материалы</a>

<a class="navbar-brand" href="news.php">Новости</a>

<a class="navbar-brand" href="cont.php">Контакты</a>

</div>

</div>

<?php
if($_SESSION['isUser'] == 0){
?>

<div id="navbar" class="navbar-collapse collapse">

<form method="post" action="index.php" class="navbar-form navbar-right">

<div class="form-group">

<input required type="text" name="login" placeholder="Email" class="form-control">

</div>

<div class="form-group">

<input required type="password" name="password" placeholder="Пароль" class="form-control">

</div>

<input type="hidden" value="1" name="enter">

<button type="submit" class="btn btn-success">Войти</button>

</form>

</div>

<?php
}

else{
?>

<div id="navbar" class="navbar-collapse collapse">

<form method="post" action="index.php" class="navbar-form navbar-right">

<h5 style="color:white;">Здравствуйтесь, <a href="kab.php" style="color:#428bca;"><?=$_SESSION['name'];?></a></h5>

<button type="submit" name="exit" class="btn btn-danger">Выйти</button>

</form>

</div>

<?php

```

```

    }
    ?>
</div>
</nav>
<div class="jumbotron">
    <div class="container">
        <h1>Добро пожаловать!</h1>
        <p>Данный ресурс реализует автоматическую проверку заданий при использовании дистанционных технологий обучения. Если вы являетесь студентом, зарегистрируйтесь!</p>
        <p><a class="btn btn-primary btn-lg" href="reg.php" role="button">Регистрация &raquo;</a></p>
    </div>
</div>
<div class="container">
    <div class="row">
        <div class="col-md-4">
            <h2>Инструкция</h2>
            <p>Инструкция по работе с системой автоматической проверки заданий при использовании дистанционных технологий.</p>
            <p><a class="btn btn-default" href="info.php" role="button">Подробнее &raquo;</a></p>
        </div>
        <div class="col-md-4">
            <h2>Методические материалы</h2>
            <p>Теоретические материалы, презентации, практические задания.</p>
            <p><a class="btn btn-default" href="mater.php" role="button">Подробнее &raquo;</a></p>
        </div>
        <div class="col-md-4">
            <h2>Примеры оформления решений</h2>
            <p></p>
            <p><a class="btn btn-default" href="cod.php" role="button">Подробнее &raquo;</a></p>
        </div>
    </div>
</div>
<hr>
<footer>
    <p>&copy; 2016 Company, Inc.</p>

```

```

</footer>

</div>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script>window.jQuery || document.write('<script src="../../assets/js/vendor/jquery.min.js"></script>')</script>

<script src="../../dist/js/bootstrap.min.js"></script>

<script src="../../assets/js/ie10-viewport-bug-workaround.js"></script>

</body>

</html>

```

Содержание файла kab.php

```

<?php
session_start();

if ($_SESSION['isUser'] == 0) {
    header('Location: index.php', true, 301);
}

?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="utf-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1">

    <meta name="description" content="">

    <meta name="author" content="">

    <title>Кабинет</title>

    <link href="css/bootstrap.css" rel="stylesheet">

    <link href="css/jumbotron.css" rel="stylesheet">

    <script src="js/ie-emulation-modes-warning.js"></script>

    <script src="js/dropdown.js"></script>

    <script src="https://code.jquery.com/jquery.min.js"></script>

    <script lang="javascript">

        function check_course(a){

            var qwe = document.getElementById('course_'+a).getAttribute('data-type');

            for(var i=1; i<qwe; i++){

```

```

        document.getElementById('course_'+i).style.display="none";
    }

    document.getElementById('course_'+a).style.display="table-row";

}

</script>

<link href="css/bootstrap.css" rel="stylesheet">

<link href="css/style.css" rel="stylesheet">

<link href="css/slider.css" rel="stylesheet">

<link href="css/popup.css" rel="stylesheet">

<link href="css/responsive.css" rel="stylesheet">

<script src="js/bootstrap.min.js"></script>

<script src="jquery-1.12.3.js" type="text/javascript"></script>

<script src="mask.js" type="text/javascript"></script>

</head>

<body>

<?php
include "db.php";

$id = $_SESSION['id'];

$id_group = $_SESSION['id'];

$sql1 = "SELECT * FROM `course`";

$retval1 = mysql_query($sql1, $link);

$course[][] = array();

$i=0;

while ($data = mysql_fetch_array($retval1)) {
    $i++;

    $course[$i]['name'] .= $data["name_c"];

    $course[$i]['id'] .= $data["id"];
}

$sql3 = "SELECT * FROM `results` WHERE `id_student`='$id'";

$retval3 = mysql_query($sql3, $link);

$results[][] = array();

while ($data = mysql_fetch_array($retval3)) {

```

```

    $task_id = $data["id_task"];
    $results[$task_id]['answer'] .= $data["percentage"];
}
$sql2 = "SELECT * FROM `task`";
$retval2 = mysql_query($sql2, $link);
$tasks[][] = array();
$i=0;
while ($data = mysql_fetch_array($retval2)) {
    $i++;
    $tasks[$i]["id"] .= $data["id"];
    $tasks[$i]["name"] .= $data["name"];
    $tasks[$i]["task"] .= $data["task"];
    $tasks[$i]["input"] .= $data["input"];
    $tasks[$i]["output"] .= $data["output"];
    $tasks[$i]["id_course"] .= $data["id_course"];
}
?>

<nav class="navbar navbar-inverse navbar-fixed-top">
    <div class="container">
        <div class="navbar-header">
            <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar"
                aria-expanded="false" aria-controls="navbar">
            </button>
            <a class="navbar-brand" href="index.php">Главная</a>
            <a class="navbar-brand" href="mater.php">Материалы</a>
            <a class="navbar-brand" href="news.php">Новости</a>
            <a class="navbar-brand" href="cont.php">Контакты</a>
        </div>
        <div id="navbar" class="navbar-collapse collapse">
            <form action="index.php" method="post" class="navbar-form navbar-right">
                <div class="form-group">
                    <h5 style="color: white"><?=$_SESSION['s_name'];
                    echo ' ' . $_SESSION['name']; ?></h5>

```

```

    </div>

    <button type="submit" name="exit" class="btn btn-danger">Выйти</button>

</form>

</div>

</div>

</nav>

<div class="container" style="padding-top: 100px;">

    <div class="row">

        <div class="col-xs-2">

            <div class="btn-group">

                <ul class="nav nav-pills nav-stacked">

                    <li class="active"><a href="mater.php">Материалы лекций</a></li>

                    <li><a href="lr.php">Лабораторные работы</a></li>

                    <li><a href="res.php">Успеваемость</a></li>

                </ul>

            </div>

        </div>

        <div class="col-xs-8">

            <div class="row">

                <div class="col-xs-12">

                    <div class="btn-group ">

                        <button type="button" class="btn btn-default dropdown-toggle" data-toggle="dropdown">Выберите
                            дисциплину

                        <span class="caret"></span></button>

                        <ul class="dropdown-menu" role="menu">

                            <?php
                                for ($i = 0; $i <= count($course); $i++)

                                    echo ' <li ><a onclick="check_course('.$course[$i]['id'].')"' href="#">' . $course[$i]['name'] .
'</a></li>';

                            ?>

                        </ul>

                    </div>

                </div>

            </div>

        </div>

        <div class="col-xs-12">

```

```

<div class="panel panel-default" >

  <div class="panel-heading">Задания</div>

  <table class="table">

    <thead>

      <tr>

        <th>Id</th>

        <th>Название задачи</th>

        <th>Дисциплина</th>

        <th>Описание</th>

        <th>Выполнить</th>

      </tr>

    </thead>

    <tbody>

      <?php
      for($i=1;$i<count($tasks);$i++)
      {
        ?>

        <tr style="" data-type="<?= count($tasks); ?>" id="course_<?= $tasks[$i]['id_course'] ?>">

          <td><?= $tasks[$i]['id'] ?></td>

          <td><?= $tasks[$i]['name'] ?></td>

          <td><?= $course[$tasks[$i]['id']]['name'] ?></td>

          <td><?= $tasks[$i]['task'] ?></td>

          <td>

            <?php
            if(!isset($results[$i]['answer']))
            {
              ?>

              <?=$results[$i]['answer']?>

              <a href="task.php?id=<?= $tasks[$i]['id'] ?>" type="submit"
                class="btn-xs btn-primary">Выбрать</a>

              <?php
              }
            else{

```

```

        ?>

        <h4> <?=$results[$i]['answer']?></h4>

        <?php
        }
        ?>

    </td>

</tr>

<?php
}
?>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>

<script src="../../dist/js/bootstrap.min.js"></script>

<script src="../../assets/js/ie10-viewport-bug-workaround.js"></script>

</body>

</html>

```

Содержание файла task.php

```

<?php
session_start();
if ($_SESSION['isUser'] == 0) {
    header('Location: index.php', true, 301);
}
?>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="utf-8">

```

```

<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">

<meta name="description" content="">
<meta name="author" content="">
<title>Кабинет</title>
<link href="css/bootstrap.css" rel="stylesheet">
<link href="css/jumbotron.css" rel="stylesheet">
<script src="js/ie-emulation-modes-warning.js"></script>
<script src="js/dropdown.js"></script>
<script src="https://code.jquery.com/jquery.min.js"></script>
<link href="css/bootstrap.css" rel="stylesheet">
<link href="css/style.css" rel="stylesheet">
<link href="css/slider.css" rel="stylesheet">
<link href="css/popup.css" rel="stylesheet">
<link href="css/responsive.css" rel="stylesheet">
<script src="js/bootstrap.min.js"></script>
<script src="jquery-1.12.3.js" type="text/javascript"></script>
<script src="mask.js" type="text/javascript"></script>
</head>
<body>
<?php
include "db.php";
if (isset($_GET['id'])) {
    $task_id = $_GET['id'];
}
$id = $_SESSION['id'];
$sql2 = "SELECT * FROM `task` WHERE `id`='".$task_id'";
$retval2 = mysql_query($sql2, $link);
$tasks[][] = array();
$task[] = array();
while ($data = mysql_fetch_array($retval2)) {
    $task["id"] = $data["id"];

```

```

$task["name"] .= $data["name"];
$task["task"] .= $data["task"];
$task["input"] .= $data["input"];
$task["output"] .= $data["output"];
$task["id_course"] .= $data["id_course"];
}
?>
<nav class="navbar navbar-inverse navbar-fixed-top">
  <div class="container">
    <div class="navbar-header">
      <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar"
        aria-expanded="false" aria-controls="navbar">
      </button>
      <a class="navbar-brand" href="index.php">Главная</a>
      <a class="navbar-brand" href="mater.php">Материалы</a>
      <a class="navbar-brand" href="news.php">Новости</a>
      <a class="navbar-brand" href="cont.php">Контакты</a>
    </div>
    <div id="navbar" class="navbar-collapse collapse">
      <form action="index.php" method="post" class="navbar-form navbar-right">
        <div class="form-group">
          <h5 style="color: white"><?= $_SESSION['s_name'];
          echo ' ' . $_SESSION['name']; ?></h5>
        </div>
        <button type="submit" name="exit" class="btn btn-danger">Выйти</button>
      </form>
    </div>
  </div>
</nav>
<div class="container" style="padding-top: 100px;">
  <div class="row">
    <div class="col-md-4">
      <div class="btn-group">

```

```

<ul class="nav nav-pills nav-stacked">
  <li class="active"><a href="mater.php">Материалы лекций</a></li>
  <li><a href="lr.php">Лабораторные работы</a></li>
  <li><a href="res.php">Успеваемость</a></li>
</ul>
</div>
</div>
<div class="col-md-8">
  <h2><?= $task["name"]; ?></h2><br>
  <h3>Задание:</h3><br>
  <p><?= $task["task"]; ?></p><br>
  <h4>Входные данные</h4>
  <p><?= $task["input"]; ?></p>
  <h4>Выходные данные</h4>
  <p><?= $task["output"]; ?></p><br>
  <div class="alert alert-info">Отправить решение</div>
  <table class="table">
    <tbody>
      <tr>
        <td style="width:40%">Студент</td>
        <td style="width:40%"><?= $_SESSION['s_name'];
          echo ' ' . $_SESSION['name']; ?></td>
        <td style="width:20%"></td>
      </tr>
      <tr>
        <td>Файл задачи</td>
        <td>
          <form action="files/upload.php" method="post" enctype="multipart/form-data">
            <p>
              <b>Укажите файл для отправки на сервер:</b>
              <input class="btn" name="qwe" type="file" size="30">
            </p>
            <input type="hidden" name="student_id" value="<?= $id; ?>">
          </form>
        </td>
      </tr>
    </tbody>
  </table>

```

```
<input type="hidden" name="task_id" value="<?= $_GET['id']; ?>">
</td>
<td>
<input class="btn" type="submit" value="Отправить">
</form>
</td>
</tr>
</tbody>
</table>
</div>
</div>
</div>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/jquery.min.js"></script>
<script src="../../dist/js/bootstrap.min.js"></script>
<script src="../../assets/js/ie10-viewport-bug-workaround.js"></script>
</body>
</html>
```