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»**

09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА

ПРИКЛАДНАЯ ИНФОРМАТИКА В СОЦИАЛЬНОЙ СФЕРЕ

БАКАЛАВРСКАЯ РАБОТА

на тему **Разработка генератора php-страниц для проведения онлайн-тестирования**

Студент(ка) С.В. Алёхин

Руководитель А.П. Тонких

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 Алёхин Степан Владимирович
1. Тема «Разработка генератора php-страниц для проведения онлайн-тестирования»
2. Срок сдачи студентом законченной выпускной квалификационной работы 23 мая 2016 года
3. Исходные данные к выпускной квалификационной работе:
Требования к программному изделию: генератор php-страниц для проведения онлайн-тестирования должен работать на базе web-сервера (Apache, ОС Linux) в технологии «клиент-сервер»
Режим функционирования: круглосуточно (365/24/7)
4. Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов, разделов)
Введение
Глава 1 Анализ методов реализации генератора php-страниц для проведения онлайн-тестирования
1.1 Анализ методов тестирования в сфере образования
1.2 Выбор средств структурного анализа для разработки генератора php-страниц с целью проведения онлайн-тестирования
1.3 Выбор способа реализации генератора php-страниц для проведения онлайн-тестирования
1.4 Анализ программных средств реализации генератора php-страниц для проведения онлайн-тестирования
Глава 2 Проектирование генератора php-страниц для проведения онлайн-тестирования
2.1 Разработка информационной модели
2.2 Разработка логической структуры базы данных

2.3 Формирование физических таблиц в базе данных
2.4 Проектирование графического пользовательского интерфейса
Глава 3 Реализация генератора php-страниц для проведения онлайн-тестирования
3.1 Реализация среды тестирования
3.2 Реализация генератора тестов
3.3 Тестирование системы онлайн-тестирования
Заключение
5. Ориентировочный перечень графического и иллюстративного материала: модель системы тестирования (нулевой уровень), модель системы тестирования (первый уровень), модель системы тестирования (второй уровень), логическая структура базы данных, физическая структура базы данных, схема взаимодействия web-страниц, шаблон страницы теста, шаблон страницы с результатами теста.
6. Дата выдачи задания «11» января 2016 г.

Руководитель выпускной
квалификационной работы _____ А.П. Тонких

Задание принял к исполнению _____ С.В. Алёхин

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 Алёхина Степана Владимировича
по теме «Разработка генератора php-страниц для проведения онлайн-тестирования»

Наименование раздела работы	Плановый срок выполнения раздела	Фактический срок выполнения раздела	Отметка о выполнении	Подпись руководителя
Выбор и утверждение темы ВКР	13.01.2016			
Поиск и анализ литературы по проблеме организации онлайн-тестирования	18.01.2016			
Анализ бизнес-процессов организации онлайн-тестирования	25.01.2016			
Концептуальное моделирование предметной области	15.02.2016			
Обоснование выбора средств реализации системы онлайн-тестирования	29.02.2016			
Реализация системы онлайн-тестирования	15.04.2016			
Заключение	06.05.2016			
Предзащита	27.05.2016			
Рецензирование	01.06.2016			
Сдача на кафедру ПЗ и материалов	15.06.2016			
Защита	16.06.2016			

Руководитель выпускной
квалификационной работы _____ А.П. Тонких

Задание принял к исполнению _____ С.В. Алёхин

Аннотация

Актуальность темы «Разработка генератора php-страниц для проведения онлайн-тестирования» обусловлена необходимостью разработки программной оболочки, позволяющей автоматически создавать средства проведения тестирования в режиме онлайн на основе текстового документа, содержащего структуру теста в виде групп, состоящих из задаваемых вопросов и вариантов ответа на них, включая правильный, и количества вопросов, которые будут выбраны случайным образом из этой группы.

Целью данной работы является разработка средства, генерирующего php-страницы, содержащие вопросы тестирования, и позволяющего контролировать знаний обучающихся в режиме онлайн.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Проанализировать научную и учебно-методическую литературу, необходимую для разработки генератора php-страниц с вопросами теста и написания пояснительной записки.
2. Проанализировать процесс контроля знаний обучающихся в учреждениях высшего образования и существующие методы тестирования с целью определения технологии проектирования и выявления функций, необходимых для реализации при разработке php-генератора.
3. Построить модель проведения онлайн-тестирования для выявления основных функций системы тестирования и генератора php-страниц.
4. Проанализировать существующие системы автоматизации программирования, сред разработки приложений, средств web-программирования и выбрать наиболее подходящие программные продукты для реализации системы тестирования и генератора php-страниц. Спроектировать базу данных системы тестирования с учётом всех требований к её разработке.

5. Проанализировать существующие архитектуры прикладных программных средств и выбрать наиболее подходящую для разработки системы тестирования.

6. Спроектировать интерфейс системы тестирования, посредством которого будет реализована выдача заданий и обработка ответов, и интерфейс генератора php-страниц, с помощью которого оператор будет создавать необходимый набор php-файлов.

7. Разработать генератор php-страниц для проведения онлайн-тестирования выбранными средствами реализации.

Выпускная квалификационная работа бакалавра состоит из 59 страниц. В ней: рисунков – 8, таблиц – 5, список литературы состоит из 26 источников. Она состоит из введения, трёх глав и списка использованной литературы.

Первая глава посвящена анализу современных методов тестирования в сфере образования и средств реализации системы тестирования, описанию возможностей контроля знаний с использованием современных информационных технологий.

Вторая глава посвящена проектированию генератора php-страниц посредством построения логической и физической структур базы данных, построению информационной модели системы тестирования, проектированию графического пользовательского интерфейса системы тестирования для контроля знаний обучающихся.

В третьей главе описаны основные этапы развёртывания среды тестирования и ключевые моменты разработки генератора php-страниц, представлен ход тестирования разработанного программного продукта.

Результатом работы является разработанный генератор php-страниц, предназначенный для проведения тестирования студентов высших учебных заведений в онлайн-режиме, позволяющий снизить загруженность преподавателей и затраты рабочего времени на контроль знаний обучающихся, обеспечить непрерывную обратную связь, предоставляющую информацию преподавателю об уровне обученности обучаемого.

Оглавление

Введение

Для осуществления современного образовательного процесса необходимым является наличие систематически организованной обратной связи, которая предоставляет обучающему информацию об уровне обученности каждого студента и о трудностях, которые возникают у обучаемого при обучении. Подобную связь можно установить разнообразными методами контроля. В настоящее время при организации образовательной деятельности в процесс обучения внедрены и закрепились технические средства, позволяющие автоматизировать и ускорить контролирующую функцию обучения.

Проблема разработки и освоения средства объективной проверки знаний обучающихся в образовании на сегодняшний день крайне актуальна. Её преодоление играет важную роль в связи с тем, что активное применение технических средств контроля способствует поддержке требуемого уровня обучения студентов, а также даёт возможность обучаемому посвятить большее количество времени индивидуальной работе с обучаемыми. При этом система контроля знаний не исключает обучаемого из процедуры проверки знаний, а только освобождает преподавателя от ряда формальных и трудоёмких процессов, предоставляя возможность сконцентрировать внимание на специфических личных трудностях каждого обучаемого. Следовательно, роль обучающего лишь возрастает, а круг его возможностей расширяется.

Использование компьютерных технологий позволяет организовать предварительную проверку знаний, промежуточный контроль и итоговую аттестацию, содействуя улучшению эффективности обучения.

Использование компьютера для сопровождения измерения объёма знаний с применением тестирования, рейтинга и методов математической статистики вызвано следующими факторами.

Широко применяемые формы контроля знаний, такие как экзамен по билетам, зачёт, коллоквиум, не всегда удовлетворяют условиям воспроизведения и сравнения результатов, которые могут быть получены на разных группах обучаемых, так как основываются на субъективной оценке

обучающих. Несмотря на то, что практика подтверждает наличие подобных методов оценки, всё-таки при их использовании на оценивание уровня знаний обучаемого возможно негативное влияние антипатии, снисхождения, переоценки или недооценки ответа. Поэтому полученную оценку нельзя рассматривать как объективную.

Также при использовании традиционных форм контроля имеются серьёзные трудности в случае массовых испытаний, которые связаны с значительным объёмом сведений, которые необходимо обработать за небольшой период времени.

Значительные трудности вызывает и необходимость выбора форм представления полученных сведений. Эта задача обеспечивает эффективность принятия оптимальных решений на основе анализа информации в зависимости от поставленных целей. Практически нереализуемой остаётся идея разработки индивидуального задания для каждого обучаемого, которое было бы типовым.

Отсутствует и адекватное средство самоконтроля, позволяющее оценить обучаемому свой собственный уровень и готовность к аттестации. Зачастую это служит причиной возникновения боязни контрольного мероприятия либо наоборот, завышенной оценки собственного уровня и халатного отношения к подготовке к контрольному мероприятию.

Целями диагностической функции контроля являются получение сведений об уровне готовности обучаемого, определение уровня его знаний. Для реализации этой функции в учреждении высшего образования осуществляют такие способы измерения учебных достижений обучаемых, как устная и письменная самостоятельная работа, расчётно-графическая работа, курсовое проектирование, выпускная квалификационная работа, экзамены и зачёты. Каждый из этих способов имеет свои плюсы и минусы.

В соответствии с принципом объективности традиционные методы оценки имеют ярко выраженные недостатки, к которым можно отнести субъективизм экзаменатора; различные уровни требований обучающих, что затрудняет сопоставление результатов обучаемых; особенность оценивания

преподавателем в плане большего акцентирования на точности, основательности или оригинальности ответа. При выставлении отметок экзаменатор не способен отвлечься от психоэмоциональных и личных факторов, сопутствующих процедуре оценки. Для оценки знаний обучаемых требуется их также измерить, но точная единица, которая позволила бы измерить знания обучаемых, отсутствует. Однако если за каждое правильное задание начислять один балл, полученное количество баллов получает чёткий смысл: отличия в количестве свидетельствуют об отличиях в умениях обучаемых выполнять задачи разной степени сложности. Имеются и иные способы, в которых вводится искусственная единица измерения.

Преодолеть указанные недостатки традиционных форм контроля способны разнообразные тестовые средства, которые основываются на общих принципах и критериях разработки. Для обеспечения качественного процесса обучения и организации бесконтактной работы обучаемых необходимо создать систему проверки и самопроверки знаний посредством онлайн-тестирования по различным дисциплинам.

Целью бакалаврской работы является разработка генератора php-страниц для проведения онлайн-тестирования.

Основываясь на том, что контроль должен быть объективным и предоставлять корректную информацию об уровне обученности, можно отметить, что при помощи онлайн-тестирования проще обеспечить надёжность и объективность оценивания. Онлайн-тестирование также позволяет одновременно проверить уровень знаний всех испытуемых в одинаковых условиях с заранее определёнными, одинаковыми для всех критериями оценки.

Для наполнения системы онлайн-тестирования должны быть созданы контрольно-измерительные материалы по ключевым темам дисциплин с соблюдением принципов разработки тестовых заданий с последующей их проверкой на соответствие ряду требований, к которым относят объективность, дифференцирующую способность и надёжность.

Глава 1 Анализ методов реализации генератора php-страниц для проведения онлайн-тестирования

1.1 Анализ методов тестирования в сфере образования

Современное состояние контроля знаний и навыков в системе среднего и высшего образования РФ представляет из себя сочетание нового и старого, объективного и субъективного, устного или письменного опроса с пятибалльным оцениванием и использованием технологий тестирования [1]. На современном этапе реформирования образования тестирование представляет из себя динамично развивающуюся сферу деятельности, связывая автоматизацию, метрологию, математику и педагогику. Введение и развитие тестовых методов вносит значительный вклад в реформу системы среднего и высшего образования в России, приобщая её к международным достижениям в сфере образования. Основываясь на важности методов контроля достижений обучаемых для педагогики, можно посмотреть на развитие тестовых методов и настоящее состояние сферы тестирования в РФ.

Контроль достижений при помощи разнообразных заданий используется уже на протяжении четырёх тысячелетий. Археологические свидетельства утверждают, что ещё древние шумеры использовали контроль по распознаванию текста изученных произведений различных авторов для контроля достижений обучаемых и усвоения изученного материала. Но научные работы по использованию тестов возникли лишь в начале двадцатого столетия, связав опыт педагогики, социологии, психологии и наук о поведении. Основоположниками науки о тестирования стали учёные Фрэнсис Гальтон и Джеймс Маккин Кеттел. Тестированию посвящены работы доктора педагогических наук, профессора Вадима Сергеевича Аванесова, Анны Анастази, Хорста Зиверта.

Психологи за рубежом изучают тестирование в рамках психометрической науки, а зарубежные педагоги – в рамках педагогических измерений. В отечественной литературе науку о тестировании называют тестологией, а тестом – небольшое стандартизованное испытание для проведения

объективного количественного измерения итогов обучения или для определения характеристик и личностных свойств человека, которые подвергаются изучению.

Педагогическим называют тест, который направлен на объективный контроль педагогических достижений. Он включает в себя систематизированный набор заданий специальной формы и определённого наполнения, который даёт возможность эффективно и качественно осуществить измерение уровня и предоставить оценку структуре учебных достижений испытуемых, проконтролировать итог обучения. В зависимости от поставленных целей педагогического теста различают тест на профессиональную ориентацию; тест, направленный на определённые критерии; тест, направленный на определённые нормативы; тест для прохождения аттестации; тест для прогноза итогов обучения.

Различают два вида заданий: 1) закрытые, заключающиеся в выборе одного варианта, задания на установку соотношений и задания на формирование верного порядка; 2) открытые, в постановке которых не предлагаются варианты ответа, то есть ответ должен набрать сам испытуемый.

Создание заданий теста проводят на базе комплекса обоснованных требований по установленной методике. В рамках принятой в международной практике методики, к этим требованиям относятся приведённые далее: в тексте задания нужно убрать отсутствие ясности и двусмысленность; само задание необходимо сформулировать максимально сжато, в одно предложение; варианты ответов к одному заданию теста должны иметь примерно одинаковое количество символов; из текста задания нужно убрать формулировки, которые могут способствовать определению верного ответа при помощи догадки, например однокоренные слова.

Содержание заданий и всего теста в целом должно соответствовать определённым условиям, которые задают комплекс нормативных требований к качеству материала теста: задания теста должны полностью отражать наполнение материала, знание которого проверяется; наполнение заданий теста

должно соответствовать достижениям, наличие которые проверяется тестом; задания необходимо привести в соответствие требованиям рабочей программы по дисциплине; наполнение заданий теста должно отвечать требованиям качества.

Кроме параметров, классическая психометрика определяет ключевые принципы отбора наполнения материала тестов: вариативность; сбалансированность; соответствие цели; взаимосвязь с формой; важность; системность; нарастающая сложность; репрезентативность; достоверность; комплексность; соответствие современному состоянию науки.

Измерение качества заданий основывается на корреляционной теории, ключевыми критериями которой служат валидность и надёжность. Надёжность определяется неизменностью итогов тестирования, которые получают при его использовании. Валидность определяется возможностью тестового материала качественно оценивать те знания и навыки, на которые он направлен. Определение валидности и надёжности заданий производится путём статистического анализа итогов проведённого массового тестирования различных групп обучаемых.

Длина теста – это количественный объём тестовых заданий, которые выдаются испытуемому. В соответствии с классической тестологией, чем больше длина теста, тем надёжнее тест. Однако опыт тестирования свидетельствует, что если в тесте имеется большое количество заданий, то снижаются такие важные показатели, как внимание и мотивация. Наилучшее количество заданий – от тридцати до шестидесяти. Тест имеет наилучшее время тестирования – время от запуска тестирования до времени наступления утомления. Диапазон времени начала утомления имеет значительные колебания – от двадцати до ста минут у лиц с одинаковым возрастом. Главным причинами утомления являются специфические особенности обучаемых, к которым также относятся мотивация и возраст.

Наименьшее время, отводимое на тест, находится в зависимости от формы, объёма и трудности заданий. К примеру, чтобы выполнить закрытое

задание с выбором одного варианта из имеющихся достаточно десяти-пятнадцати секунд.

Педагогические тесты стремительно приобрели популярность в развитых государствах в начале двадцатого века. Тем не менее, в Советском Союзе к тридцатым годам прошлого века была развёрнута их полномасштабная критика, а позже они были полностью запрещены. Лишь в начале девяностых годов прошлого столетия педагогические тесты стали вводиться в системы среднего и высшего образования РФ. На сегодняшний день тестирование в РФ развивается в двух направлениях: на федеральном уровне Министерство образования реализует проекты ЕГЭ, Основной государственный экзамен; на уровне корпораций теория тестов стремительно развивается в рамках учреждений высшего образования.

На уровне корпораций научными разработками в области образовательных измерений и проведением тестов в РФ занимается множество разных организаций. Среди учреждений высшего образования, активно занимающихся тестированием и развивающих это направление, можно выделить Современную гуманитарную академию (СГА), Тольяттинский государственный университет, Национальный исследовательский ядерный университет МИФИ и прочие учреждения высшего образования. В СГА на базе психометрики были разработаны подходы к выявлению темпов и уровней достижений обучаемых, на базе теории линков, которые представляют собой единицы знаний, и физиологических процессов. Также в СГА для выявления усвоения новых терминов и навыков обучающихся используются интеллектуальные образовательные программы – супертьюторы, в которых запрограммирована обратная связь.

С развитием web-технологий появилась возможность проведения онлайн-тестирования с применением локальных сетей и сети Интернет. Онлайн-тестирование для измерения качества обучения предоставляет набор преимуществ перед применением традиционных методов контроля достижений обучаемых. Это возможность проведения централизованного контроля,

который даёт возможность обеспечить охват всей необходимой категории обучающихся. Также широкое внедрение компьютерных технологий предоставляет возможность организовать максимально объективный контроль достижений, независимый от субъективности экзаменатора.

Использование компьютерных технологий способствовало созданию и широкому применению разнообразных программ для тестирования [15].

Анализ тестологической литературы и интернет-источников даёт возможность выявить главные показатели современного программного комплекса для проведения тестирования.

Главное требование к современному программного комплексу для проведения тестирования состоит в отсутствии привязки к уровню сложности, наполнению, предметной ориентированности, виду и тематике каждого задания. Такая унификация предоставляет возможность не создавать для проведения нового теста и обработки результатов проведённого тестирования новую программный комплекс, привлекая специалистов в сфере создания программных средств, а, освоив определённый программный комплекс, наполнять его новым содержанием по разным дисциплинам, руководствуясь единой методикой. Отсутствие привязки программного комплекса для проведения тестирования к конкретному материалу, выбранному для разработки теста, свидетельствует о наличии свойства универсальности разрабатываемого комплекса.

Современный программный комплекс для проведения тестирования должен включать следующие компоненты: компонент, предназначенный для разработки тестов, то есть для формирования базы заданий, стратегий проведения тестирования и измерения результатов; компонент, предназначенный для проведения тестов, то есть выдачи заданий, обработки вводимых ответов; компонент, предназначенный для мониторинга качества достижений обучаемых на протяжении всего изучения раздела дисциплины или всей дисциплины на базе ведения протокола хода тестирования, фиксирования его итогов и сохранения протоколов в систематически обновляемой базе.

Свойство модульности разрабатываемого программного комплекса для проведения тестирования заключается в наличии связанных, но при этом независимых модулей системы: модуля разработки теста, модуля протоколирования результатов, модуля проведения теста.

Свойство централизованности разрабатываемого программного комплекса для проведения тестирования заключается в том, что единую базу данных необходимо хранить на удалённом сервере.

Свойство защищённости заключается в том, что разрабатываемый программный комплекс для проведения тестирования должен разделять права пользователей по заданным ролям (обучаемый, обучающий), например, для исключения доступа обучаемых к базе с верными ответами.

Свойство адаптивности заключается в том, что разрабатываемый программный комплекс для проведения тестирования должен предоставлять возможность проведения диагностики с применением разных моделей диагностирования для получения результатов, определённых главенствующей идеей диагностики.

Свойство обработки результатов тестирования заключается в том, что разрабатываемый программный комплекс для проведения тестирования должен проводить математическую обработку результатов тестирования, в частности, определение степени трудности заданий.

Также современный программный комплекс для проведения тестирования должны характеризовать такие показатели как режимы тестирования, различные виды заданий, случайная выдача заданий и перемешивание вариантов ответа, возможность ограничения времени теста, импорт тестового материала из файлов популярных редакторов.

1.2 Выбор средств структурного анализа для разработки генератора php-страниц с целью проведения онлайн-тестирования

Для разработки генератора php-страниц в целях проведения онлайн-тестирования необходимо использовать модель, основой для которой служат ведущие стандарты управления (MRP II, ISO 9001:2015). Такая модель

создаётся методами проектирования и анализа. Научные исследования в сфере проектирования и анализа позволили разработать широкий набор программных продуктов для автоматизации программирования, которые включают в себя средства проектирования высокого уровня, а по своему качеству и лёгкости распространения не уступают уровню программных комплексов, предназначенных для технологических систем. В настоящий период своего развития программные продукты для автоматизации программирования внедрены в сферу промышленного производства прикладных программ. Они нашли применение и в решении таких проектных задач как публикация отчётов, тестирование, планирование и контроль, моделирование программных продуктов для оперативного и стратегического планирования и управления ресурсами [13].

Программные продукты для автоматизации разработки поддерживают разнообразные технологии программирования: от простейших программных продуктов для анализа и подготовки документации до сложных программных продуктов для автоматизации полного жизненного цикла.

Самыми трудоёмкими стадиями создания программ являются стадии проектирования и анализа, при осуществлении которых средства автоматизации программирования предоставляют возможность получить качество принятых решений и подготовить отчёты [3]. Значительная роль предоставлена методам отображения данных: создание диаграмм, применение разнообразных наборов цветов, контроль синтаксиса. Графическое моделирование предоставляет программистам возможность представить готовый программный комплекс, перестраивать его исходя из заданных целей и ограничений.

Под категорию средств автоматизации разработки программ подпадают достаточно дешёвые инструментальные средства для настольных компьютеров и дорогие мощные комплексы программ для крупных вычислительных систем. Среди программных продуктов присутствует более трёх сотен разнообразных

средств автоматизации разработки программ, которые применяются большинством крупных зарубежных компаний.

Средством автоматизации разработки программ можно считать любую программу, позволяющую автоматизировать стадии жизненного цикла программы и обладающую возможностями описания с помощью компьютерной графики и составления отчётов, возможностями предоставления инструментов комфортной разработки, обогащающих возможности программиста; совместимостью изолированных элементов автоматизации, которая обеспечивает лёгкость управления процедурами разработки программ; возможностью применения особого хранилища данных.

Программные продукты для автоматизации разделяют по типам. Тип определяется направленностью программного продукта на определённые стадии жизненного цикла программы. Разделение средств автоматизации на категории отражает выполняемые средством функции. При делении на категории выделяют разрозненные локальные средства автоматизации, выполняющие отдельные независимые задачи; продукты с частичной интеграцией и средства с полной интеграцией, которые привязаны к общему хранилищу. Программные продукты для автоматизации программирования разделяют по используемым моделям данных и методам; по мере связанности с СУБД и по операционным системам.

В соответствии с разделением по типам, которое соответствует структуре средств автоматизации программирования, выделяют программные продукты для составления модели; широко распространённые программные продукты для проектирования и анализа, которые поддерживают методы разработки проектов (например, Designer от компании Oracle); программные продукты для составления модели данных для распространённых СУБД (к примеру, AllFusion ERwin Data Modeler); программные продукты для языков четвёртого поколения (например, Developer Suite от компании Oracle, Team Developer); программные продукты для генерации кода; программные продукты для получения модели по коду, которые анализируют код и формируют модели на его основе;

программные продукты для формирования диаграмм ERD (входят в состав Oracle Designer, AllFusion ERwin Data Modeler).

В сфере анализа кода распространённость получил программный продукты для автоматизации компании IBM Rational Rose, предназначенный среди прочего и для подготовки отчётов.

Этот продукт применяет синтез-методы, которые основаны на подходах трёх разработчиков универсальной нотации UML: Гради Буча, Джеймса Рамбо и Айвара Джекобсона. Нотация, предназначенная для получения моделей объектов, фактически задаёт стандарты в сфере проектирования и анализа [12]. Необходимая версия продукта Rational Rose зависит от среды, в которой пишется программа (Ада, Team Developer). Версия для среды C++ создаёт отчёты в форме диаграмм и код на языке C++. Помимо этого, программный продукт от компании IBM даёт возможность применять разработанные программные модули в других проектах.

Работа Rational Rose заключается в создании диаграмм, которые определяют статические и динамические свойства модели.

В Rational Rose выделяют хранилище, пользовательский интерфейс, инструменты просмотра и контроля проекта, инструменты сбора статистики, инструменты генерации отчётов. В версии для C++ добавлены инструменты автоматического создания и анализа кода на C++, которые отвечают за реинжиниринг.

Хранилище – это объектная база данных. Инструменты просмотра проекта дают возможность навигации: переключение по иерархиям классов и подсистем проекта, быстрое перемещение от одного вида диаграмм к другому. Инструменты контроля и сбора статистики дают возможность убрать ошибки уже на стадии развития проекта. Инструмент создания документации создаёт тексты отчётов на основе данных из хранилища.

Инструмент генерации кода на C++ создаёт файлы заголовков и описаний классов и объектов исходя из моделей проекта. Генерируемая основа программы затем детализируется путём непосредственного программирования

на C++. Инструмент анализа кода на C++ выполнен как изолированный компонент. Он необходим для формирования модулей на основе задаваемого пользователем кода на C++. Во время проектирования инструмент анализа контролирует синтаксис кода и диагностирует ошибки, формируя модель для применения в других проектах.

Инструмент анализа C++ позволяет задать расширения исходных файлов, основной компилятор, определить, какие данные необходимо включить в разрабатываемую модель и какие компоненты результирующей модели нужно вывести на экран. Поэтому созданные в версии для C++ программные модули можно использовать повторно.

Итогом проектирования являются различные диаграммы, спецификации, заготовки кода, модель проектируемого продукта.

Модель проектируемого продукта представляет собой текстовый файл, который содержит информацию о проекте, основываясь на которой можно восстановить все диаграммы и спецификации.

Код в виде файлов типа .h (заголовки, которые описывают классы) и .cpp (заготовки для описания методов разработчиками), генерируется в определённую пользователем папку.

Версия для C++ помещает в код комментарии, начинающиеся с символов //##. Генерируемый код задаётся программой или может быть определён пользователем. Затем такой код развиваются разработчиками в готовый программный комплекс.

Чтобы обеспечить групповую работу в версии для C++, осуществляют декомпозицию модели. Образованные в результате декомпозиции подмодели можно отдельно сохранить на жёстком диске или загрузить в новую модель. Подмоделью является подсистема или категория классов.

Подмодель загружают в память, выгружают из памяти, сохраняют в виде файла, устанавливают запрет на изменения; заменяют на другую.

При работе в версии для C++ с инструментами управления конфигурацией и контроля версий можно установить запрет на изменения всех

подмоделей, кроме тех, с которыми работает данный программист. При этом для содержащих подмодели файлов, устанавливают запрет на запись. Это даёт возможность при чтении подмоделей другого разработчика сохранить запрет на изменения и исключить непреднамеренные воздействия.

Версия для C++ может работать в разных вычислительных системах: на IBM в операционной системе Windows, на SPARCstation, на Hewlett-Packard, на IBM System p.

Для работы версии для C++ необходим процессор Intel Pentium III 800 мегагерц, память 2 Гбайт (рекомендовано 3 Гбайт), свободное дисковое пространство объёмом 1 Гбайт и место для хранения проектов.

Проводить сравнение отдельно взятых инструментов автоматизации разработки программ нецелесообразно, ни один инструмент автоматизации программирования не реализует все задачи создания и сопровождения программного продукта. Это может подтвердить полный комплекс показателей выбора и оценки, которые затрагивают все стадии жизненного цикла. Сравнивать можно системы согласованных инструментов, осуществляющие поддержку полного жизненного цикла программного продукта и имеющие техническую и методическую поддержку от компаний-производителей. Объектный инструмент автоматизации программирования Rational Rose является компонентом наиболее развитого из всех поставляемых в РФ подобных комплексов – комплекса инструментов создания программных продуктов на базе методологии и технологии DATARUN.

1.3 Выбор способа реализации генератора php-страниц для проведения онлайн-тестирования

Информационные технологии в сфере образования на настоящий момент достигли такого уровня развития, на котором стоит вопрос разработки многопользовательских систем обучения и тестирования.

Различают следующие подходы к использованию прикладных программ: мейнфреймы, клиент/файловый сервер и клиент-серверный подход. Наиболее актуальным является клиент-серверный подход, который объединяет плюсы

вычислительной мощности мейнфреймов с удобством использования и низкой ценой систем на основе файл-серверного подхода. Проектируемая система тестирования должна быть организована на основе клиент-серверной архитектуры со следующими ключевыми уровнями работы с данными. Клиент обеспечивает интерфейс пользователя. Требование к вычислительной системе на стороне клиента – наличие веб-браузера. Форма с заданием и вариантами ответов будет появляться как веб-страница в окне веб-браузера. Задания можно представлять в форме текста с использованием средств компьютерной графики [8]. Испытуемый курсором мышки осуществляет выбор одного варианта ответа, после чего информация отправляется на веб-сервер.

Такой подход упрощает поддержку прикладных программ, так как структуру данных можно менять изолированно от логики их представления; каждая клиентская система может быть оптимизирована автономно в зависимости от реализуемых ею задач; до минимума снижается количество повторений кода; наличие распределённой обработки информации предоставляет возможность получения максимальной производительности и обеспечивает единовременную работу большого количества пользователей [24].

Показателями сетевой программы являются связанные между собой производительность, надёжность, интенсивность трафика сети, загруженность сетевого администратора, безопасность и баланс нагрузки.

При классическом клиент-серверном подходе клиентская часть представлена прикладной программой, созданной с помощью средств разработки приложений типа Microsoft Visual Basic или Delphi, называемой толстым клиентом. Своё наименование толстый клиент получил по тому, что кроме осуществления функции представления данных он отвечает ещё и за логику решения задачи. Толстый клиент взаимодействует и с реляционной базой данных, и с файловым сервером.

Несмотря на это, на основе клиент-серверной архитектуры можно логику прикладной задачи частично перенести на серверы, приблизив полученную архитектуру программного продукта к веб-ориентированному подходу.

Для сравнения производительности клиент-серверного и веб-ориентированного подходов сначала тестировался толстый клиент. Засекалось время реагирования интерфейса пользователя на стороне клиента, который отражает логику программы. Затем засекалось время реагирования интерфейса на стороне сервера, причём прикладные программы работали на нескольких компьютерах.

При проведении эксперимента для запуска прикладных программ использовался файловый сервер, так как в архитектуре корпоративных сетей прикладные программы хранятся на дисках файловых серверов, а не на локальных жёстких дисках. Поэтому прикладные программы порождали увеличение трафика сети при их запуске и при доступе к файлам на сервере.

Сетевая активность веб-приложений имеет другую структуру, так как сеть нагружается при обращении пользователя к веб-документу или к какой-либо части прикладной программы на веб-сервере. Помимо этого, увеличение сетевого трафика возможно при передаче информации со стороны веб-браузера на веб-сервер и при обращении сервера, на котором находятся прикладные программы к системе управления базами данных.

Значительную роль играют и применяемые стеки протоколов. Стек протоколов NetBEUI максимально нагружает сеть, стек протоколов IPX/SPX показывает умеренную сетевую активность, а самым «лаконичным» оказался стек протоколов TCP/IP. К тому же, оказалось возможным снизить трафик сервиса доменных имён DNS путём применения вместо доменного имени сервера его IP-адреса. Однако для администраторов сети использование такого метода приводит к лишней работе при подключении к сети новых компьютеров [16].

На администраторов сети обычно возлагается обеспечение бесперебойной работы серверных и клиентских прикладных программ, обеспечение защиты информации, обновление программного обеспечения.

Администратор клиент-серверной системы должен осуществлять контроль за пользованием файл-сервером, настройку логических дисков и

конфигурацию прав доступа к серверу. Периодически он обязан устанавливать новые версии программного обеспечения файл-серверов, а иногда и нести ответственность за внезапно возникшие сбои в работе прикладных программ. Он также отвечает за защиту информации, что включает в себя присвоение имён и паролей пользователям, непрерывную проверку журнала протоколирования работы сети и применение инструментов защиты, используемых в компании.

Веб-администратор выполняет практически такие же задачи. Но на работу с пользователями по перезагрузке зависающих прикладных программ ему требуется значительно меньше времени, так как браузеры «зависают» намного реже программ, которые выполнены с использованием сред разработки приложений [25].

Веб-архитектура в целом надёжнее, чем клиент-серверный подход. Необходимость сопровождение приложения в виде добавления новых атрибутов или модификация имеющихся, свидетельствуют о меньшей стабильности программного обеспечения на основе клиент-серверного подхода. Термин «толстый клиент» фактически означает «сложный». Стадии отладки и тестирования клиент-серверных прикладных программ – достаточно трудоёмкая процедура, так как такие программы включают значительное количество взаимосвязанных модулей [19].

Однако если убрать логику в изолированную процедуру сервера, то значительно облегчится сопровождение программ. К тому же, для модификации логики в архитектуре «клиент-сервер» требуется больше серьёзных затрат.

Веб-приложение снижает сложность программного обеспечения, повышает производительность и предоставляет возможность применять новейшие информационные и коммуникационные технологии, такие как сеть Интернет [10].

Веб-ориентированный подход даёт значительные преимущества перед клиент-серверным. Во-первых, за счёт интуитивно понятного интерфейса

обеспечивается возможность разделения информации из множества разных источников без изучения применяемых технологий. Во-вторых, системы можно поставлять широкому кругу пользователей, относительно недорого и с минимальными усилиями. В-третьих, применение сетевых технологий исключает необходимость в установке прикладной программы на каждом компьютере пользователя. В-четвёртых, любые модификации распространяются автоматически [14]. Именно веб-архитектура станет основой построения системы тестирования.

Исходя из выше перечисленного, можно сформулировать основные требования к системе онлайн-тестирования. Это наличие веб-архитектуры, доступность (возможность использования в любое удобное время в любом месте, где имеется доступ к сети Интернет), простой и удобный интерфейс. Разрабатываемый генератор php-страниц должен обеспечивать возможность быстрого создания и изменения веб-страниц.

1.4 Анализ программных средств реализации генератора веб-страниц для проведения онлайн-тестирования

Веб-документы разрабатываются на основе языка HTML и его расширенных версий [7]. Браузер интерпретирует команды HTML и отображает на экран устройства документ в удобной для человека форме. HTML (в переводе с английского язык гипертекстовой разметки) – это стандартизированный язык разметки веб-документов в сети Интернет [4]. Язык HTML можно использовать при создании оболочки системы онлайн-тестирования.

Для проведения же самого тестирования необходима интерактивность веб-страниц. Испытуемый должен вносить ответы в некую форму, которая затем должна подвергнуться анализу. Вносимые испытуемым данные можно подвергнуть анализу специальными серверными приложениями на языках PHP или Perl [18].

PHP – язык, который предназначен для разработки веб-страниц. PHP поддерживается многими компаниями, предоставляющими услуги хостинга

[22]. PHP включён в состав типичного комплекта инструментов веб-разработчика наряду с Linux, эмулятором Apache, СУБД MySQL [2]. Иногда место PHP занимают Perl или Python. Создатели PHP – это программисты-энтузиасты, занимающиеся расширениями языка PHP, смежными проектами и документированием языка [5].

В сфере сетевого программирования распространёнными скриптовыми языками являются PHP и JavaServer Pages из-за своей простоты, высокой скорости работы, широкой функциональности и наличия большого количества исходных кодов. Отличием PHP является наличие расширений для работы с базами данных, pdf-файлами. Каждый программист имеет возможность создать разработанное им расширение PHP. Имеются сотни php-расширений, но в стандартный дистрибутив включены немногие. Интерпретатор языка PHP подключается к серверу через модуль, разработанный специально для этого сервера или в качестве CGI-программы [20]. PHP также применяется для осуществления задач администратора во всех популярных операционных системах. Но в этом качестве он не завоевал большой популярности, отдав приоритет Visual Basic Scripting Edition. PHP применяется огромным числом программистов [6]. Синтаксис PHP аналогичен синтаксису языка C. Ряд элементов заимствованы из языка Perl [26].

Главным отличием языка Perl являются его обширные возможности работы с текстом. Perl – инструмент программирования, который применяется для огромного количества проектов в различных сферах и активно применяется при разработке сетевых прикладных программ, удовлетворяющих самые разнообразные потребности.

Однако последнее время язык Perl постепенно выходит из моды в связи с проблемами предоставления доступа. В большинстве случаев Perl работает посредством CGI, что не очень удобно. А в языке PHP никаких проблем с доступом нет.

JS – скриптовый язык, применяющийся для создания сценариев поведения веб-браузера. JS применяется для создания эффектов визуализации (меню, кнопки, подсветка).

ASP (в переводе с английского «активные страницы сервера») – технология Microsoft для разработки прикладных интернет-программ. ASP представляет собой не язык программирования, а только технологию обработки, которая предоставляет возможность подключать модули программы во время процедуры создания веб-документа. Распространённость ASP основывается на простоте применяемых языков сценариев [23]. Технология ASP развилась в новую технологию создания веб-программ ASP.NET. Недостатком является дорогой хостинг для ASP на платформе Win32 [21].

Выбор программного средства реализации системы онлайн-тестирования находится в зависимости от платформы. PHP поддерживается операционными средами семейства UNIX, известными своей стабильностью. PHP достаточно прост и имеется много примеров php-программ. К его плюсам также относят следующее. Синтаксис PHP и C++ одинаковы. В PHP удобно работать с cookies. Все передаваемые параметры HTTP автоматически преобразовываются в переменные. PHP легко вставляется в веб-страницу между командами языка HTML подобно JavaScript [9].

В качестве сервера был выбран Apache, так как связка PHP+ Apache хорошо работает в среде Windows, дистрибутивы Apache свободно распространяются [11]. Сервер Apache является самым распространённым HTTP-сервером в Интернете с апреля 1996 года, в январе 2007 года он работал на 59 % всех HTTP-серверах [17].Ключевыми достоинствами Apache являются надёжность и удобство конфигурации. Apache даёт возможность подключать внешние модули, применять системы управления базами данных, изменять сообщения об ошибках. Помимо этого, он поддерживает IPv6. Apache создаётся и сопровождается открытой разработчиками-энтузиастами при содействии ASF и входит в состав многих программных продуктов, среди которых Oracle и WebSphere.

В качестве среды разработки был выбран C++ Builder XE4.

C++ Builder (по-русски говорят [би́лдэр]) – среда разработки приложений на языке C++, которая включает библиотеку визуальных компонентов VCL, поддерживаемую компанией Embarcadero. C++ – язык общего назначения, поддерживающий различные подходы к программированию: процедурный, обобщённый, функциональный и объектный.

В девяностых годах прошлого столетия язык завоевал большую популярность среди языков общего назначения.

Язык появился в начале восьмидесятых годов прошлого века, когда сотрудник компании «Bell Laboratories» создал группу расширений языка C для личных целей. До официальной стандартизации язык развивался силами Бьёрна Страуструпа. В 2011 году был ратифицирован последний на сегодня международный стандарт языка C++: ISO/IEC 14882:2011.

К достоинствам языка C++ относят масштабируемость (имеется возможность создания приложений для самых разнообразных систем и платформ); низкоуровневое программирование (впрочем, при неаккуратном применении это преимущество стать недостатком); разработку общих алгоритмов для различных структур и вычисления с использованием шаблонов на стадии компиляции.

Таким образом, для реализации системы тестирования будет использован веб-интерфейс и язык PHP. Для реализации генератора тестов выбраны язык C++ и среда C++ Builder 10.

Глава 2 Проектирование генератора php-страниц для проведения онлайн-тестирования

2.1 Разработка информационной модели проведения онлайн-тестирования

Разработка информационной модели системы включает проектирование структур баз данных и их связей. Программный комплекс должен работать в связке с базой данных, используемой для хранения всей необходимой информации (такой как тексты вопросов и задач, структуры тестов и т. д.).

Модель контроля знаний студентов представлена на рисунке 2.1. Тесты составляются исходя из имеющегося набора материалов по дисциплине. При этом контроль должен соответствовать требованиям государственных образовательных стандартов высшего профессионального образования (ГОС ВПО) к обязательному минимуму содержания основной образовательной программы. Исходя из требований для конкретной специальности строится тематическая структура аттестационных педагогических измерительных материалов (АПИМ). Для составления тестов важно знать требования к измерительному материалу. При формировании оценки необходимо руководствоваться предварительно разработанными критериями оценки.

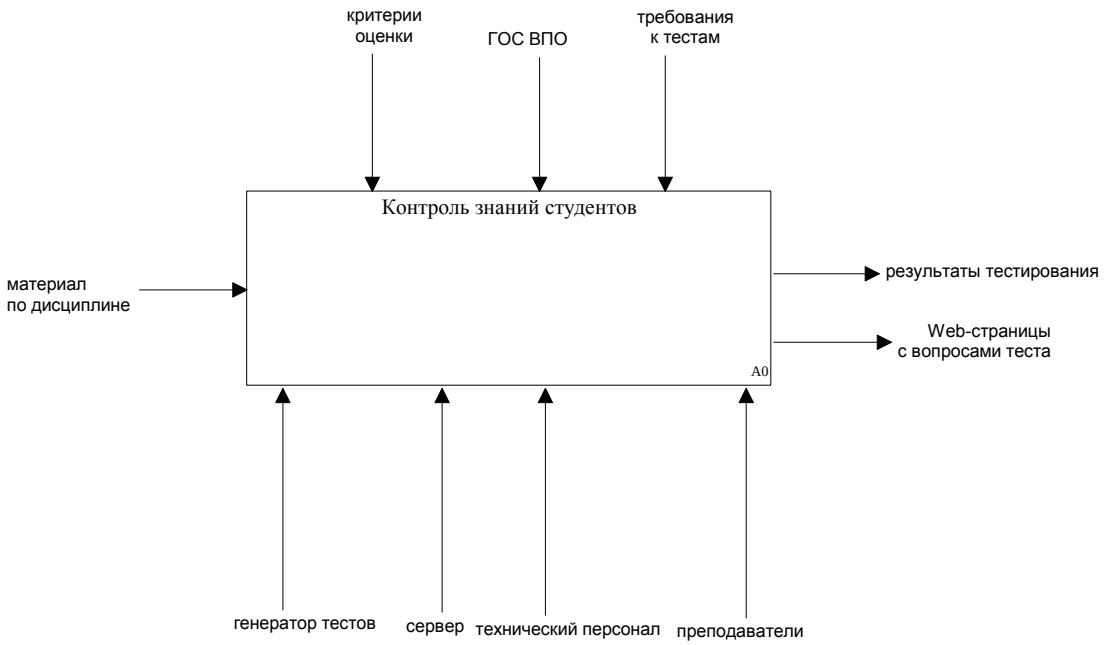

Рисунок 2.1 – Модель системы тестирования (нулевой уровень)

Сервер, на котором функционирует проектируемая система, должен выдавать Web-страницы с заданиями тестирования и формировать результаты тестов. Тесты, сформированные через генератор тестов из базы данных, составленной преподавателем, публикуются с помощью технического персонала.

Сначала преподаватель или технический персонал на основе рекомендаций преподавателя создаёт файл базы данных теста (*.txt) из имеющегося материала по дисциплине в соответствии с правилами оформления базы данных, содержанием дисциплины согласно государственным образовательным стандартам высшего профессионального образования и требований к тестам (рисунок 2.2).

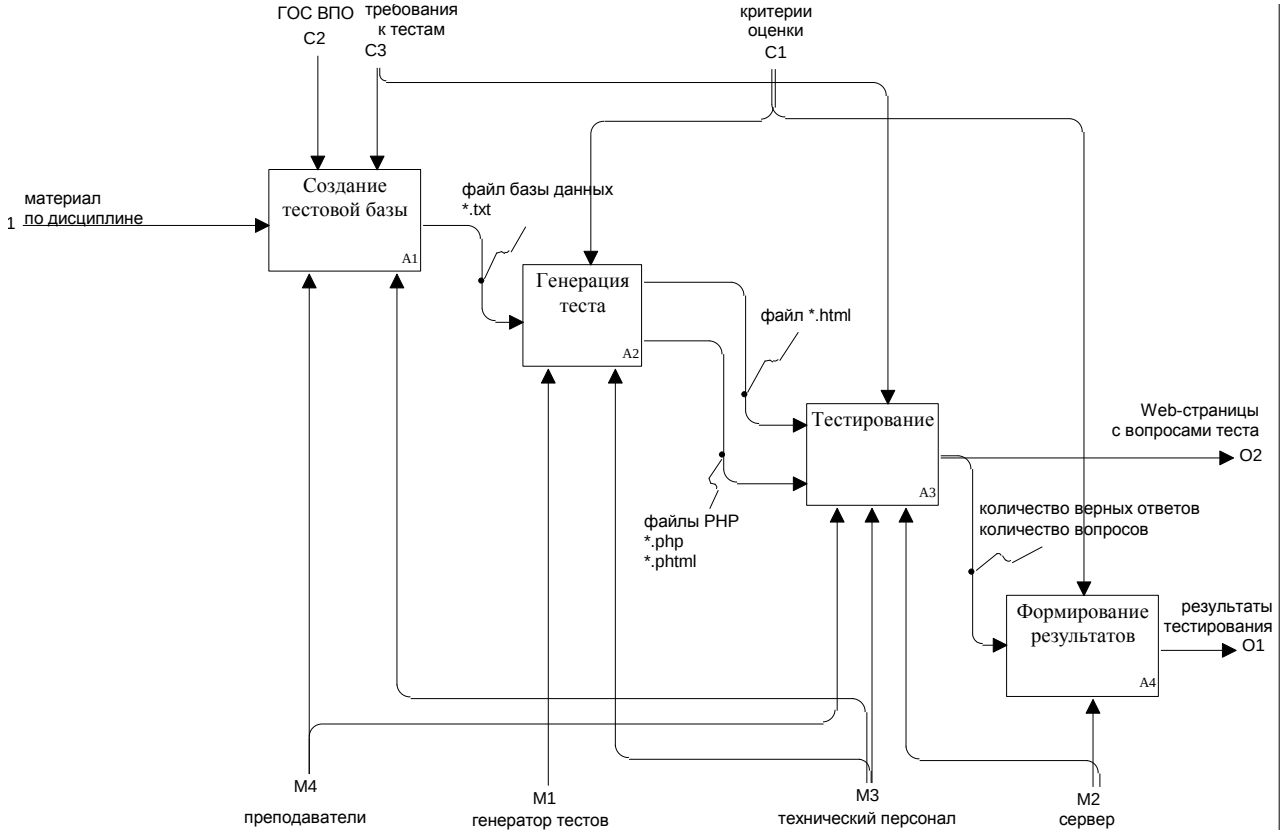

Рисунок 2.2 – Модель системы тестирования (первый уровень)

Затем с помощью генератора тестов технический персонал (инженер, техник или программист) создают файлы Web-страниц *.php и *.phtml из исходного текстового файла базы данных. Для генерируемых файлов определяется вид оценки и правила формирования оценки по результатам тестирования в соответствии с согласованными с преподавателем и

утверждёнными критериями оценки. На этом этапе также создаётся файл стартовой Web-страницы (index.html) с ссылкой на сгенерированный тест (папка\tst001.html).

Студент загружает с сервера файл начальной Web-страницы и выбирает необходимый тест (рисунок 2.3). При щелчке по выбранной ссылке осуществляется переход на первый вопрос тестирования (файл tst001.phtml). После выбора ответа на каждый вопрос автоматически с сервера открывается страница со следующим вопросом. Преподаватель или техперсонал контролируют самостоятельное выполнение студентом теста и фиксируют полученный им результат. Каждый раз после выбора студентом варианта ответа количество заданных вопросов увеличивается на один. Если ответ оказался верным, то инкрементируется и количество правильных ответов. Сервер выдаёт страницу с полученными числами и в соответствии с заданными критериями выводит оценку.

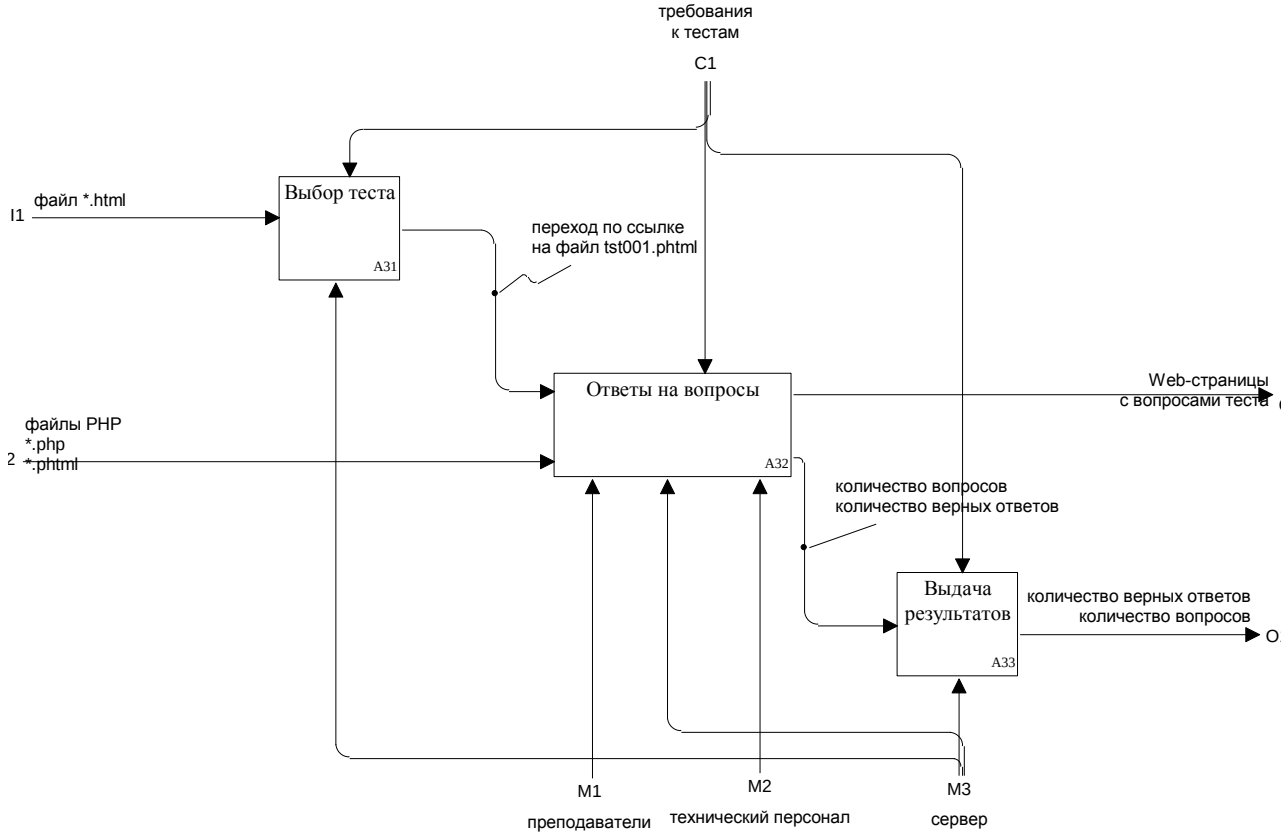

Рисунок 2.3 – Модель системы тестирования (второй уровень)

Предполагается, что учебный курс включает несколько разделов, в каждый из которых входит несколько тем, и каждый из вопросов для тестирования относится к определенной теме определенного раздела курса.

Вопросы для вариантов теста выбираются из базы данных по заложенному для данного вида теста алгоритму (определяемому структурой теста) на основе случайной выборки нужного количества вопросов из групп равноценных по уровню сложности вопросов.

Получаемые в результате выборки варианты одного и того же теста всегда различны по составу входящих в них вопросов, хотя охватывают один и тот же тематический материал и имеют одинаковый уровень сложности. Таким образом для каждого тестируемого формируется индивидуальный вариант теста.

Внутри каждого варианта теста вошедшие в него вопросы дополнительно переставляются в случайном порядке, что исключает соотнесение номеров вопросов с порядком тем учебного курса в случае если в каждой группе тестов более одного вопроса. Если студент выполняет итоговый тест, то порядок вопросов аналогичен списку дидактических единиц дисциплины.

Структура теста задаёт алгоритм формирования теста, определяющий каким образом будут выбираться вопросы для вариантов данного теста из всего множества вопросов, содержащихся в базе данных.

Формирование новых структур тестов осуществляется преподавателем с помощью программы генерации тестов. Программа позволяет задать нужное количество шагов структуры, на каждом из которых выделяется группа равноценных по уровню сложности вопросов из базы данных и задается количество вопросов теста, которые будут выбираться на основе случайной выборки из имеющейся группы вопросов.

Структура и содержание теста хранится в файле с определённым именем и расширением *.txt (файл базы данных). Каждый тест имеет название в соответствии с наименованием дисциплины, модуля или дидактической единицы. Оно будет отображено в заголовке веб-страницы при прохождении

теста и выводиться на странице результата. Файл базы данных теста также должен содержать информацию об авторе.

Тест по дисциплине включает вопросы по разным модулям, на которые разбит курс (слайд №9). Для каждого модуля задаются номер, название и количество заданий в нём. Обычно именно модуль выступает структурной единицей для теоретического материала, практических и индивидуальных домашних заданий.

Уровнем ниже находятся темы модуля. Для базы данных – это группа заданий. У неё есть название, количество заданий в группе, количество заданий, которые необходимо решить из группы.

Для успешной сдачи итогового тестирования студенты могут проходить сначала тесты по каждой дидактической единице. Такие тесты могут включать в себя все вопросы из базы данных или значительную их часть. Это исключит возможность угадывания ответа и позволит усвоить технологию решения данных заданий. Тесты по одной теме могут включать до 20 вопросов, которые могут повторяться у разных студентов, но в случайном порядке.

Затем студент может пройти итоговый тест по модулю (на закрепление имеющихся знаний, умения и навыков), содержащий по 2-5 вопросов по каждой дидактической единицы. Эти задания уже знакомы студентам. Трудности заключаются в их чередовании.

На четвёртом уровне содержатся сами задания. Каждое из заданий содержит вопрос, правильный вариант ответа и неправильные варианты.

2.2 Разработка логической структуры базы данных системы онлайн-тестирования

Качество разработанной базы данных всецело зависит от качества выполнения отдельных этапов её проектирования. Огромное значение имеет качественная разработка логической модели базы данных, так как она, с одной стороны, обеспечивает адекватность базы данных предметной области, а с

другой стороны, определяет структуру физической базы данных и, следовательно, её эксплуатационные характеристики.

В графической диаграмме схемы базы данных (рисунок 2.4) вершины графа также используются для интерпретации типов сущностей, а дуги – типов связей между типами сущностей. При реализации каждая вершина графа представляется совокупностью описаний экземпляров сущности соответствующего типа.

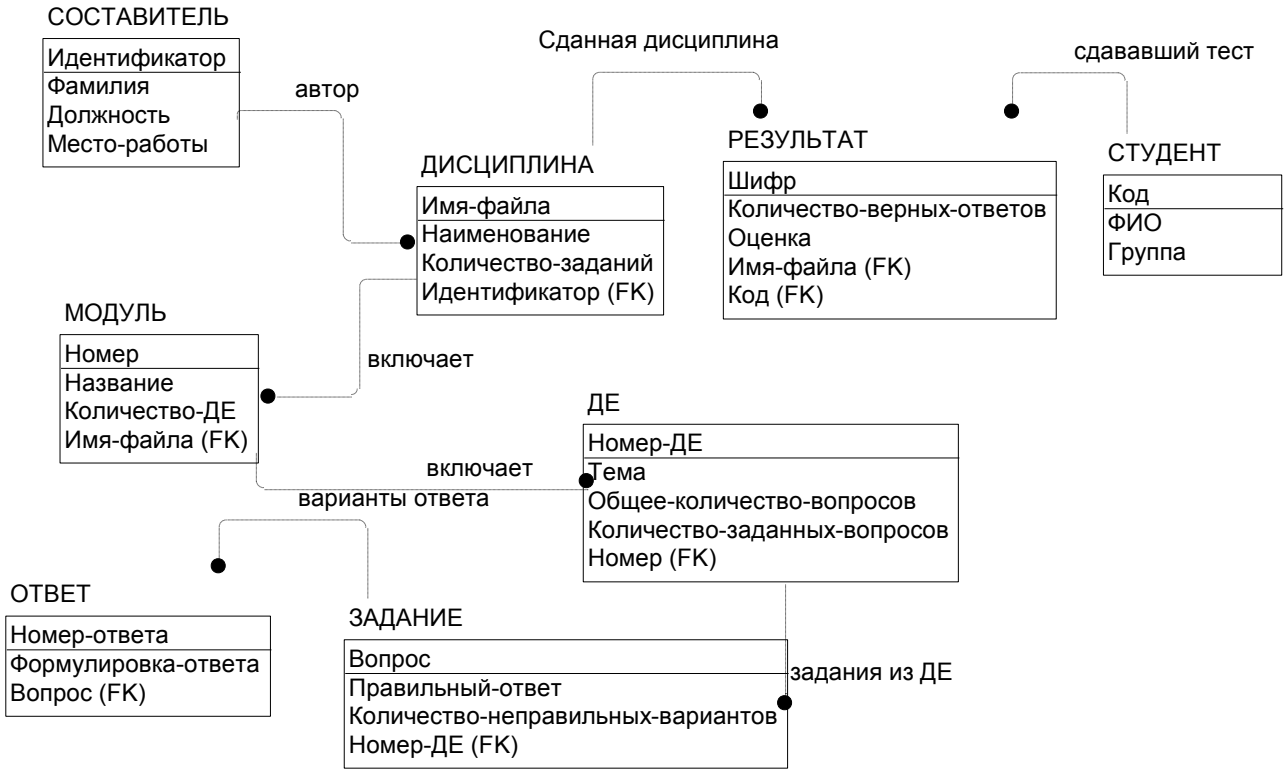

Рисунок 2.4 – Логическая структура базы данных системы онлайн-тестирования

Информация о студенте, который проходит тест, может включать код студента (уникальный идентификационный номер), фамилию, имя и отчество студента, а также название группы. Эти данные учитываются при подведении итогов тестирования.

Результат пройденного тестирования может иметь некий шифр (свой идентификатор), количество верных ответов (общее количество заданий берётся из тестовой базы) и оценку, формируемую согласно установленным критериям.

Один и тот же студент может сдавать несколько тестов, получая каждый раз очередной результат, но один результат, который можно идентифицировать

по шифру, принадлежит к одному тесту конкретного студента. Значит, связь «результат-студент» имеет тип «один-ко-многим».

Тест должен включать информацию о составителе. Она включает в себя фамилию, должность, место работы. Её можно выделить в отдельную сущность. Связь «составитель-дисциплина» – «один-ко-многим», так как один преподаватель может составлять тесты по разным дисциплинам.

При составлении теста по всей дисциплине или теста по нескольким дисциплинам, число составителей может быть больше единицы. Для реализации таких случаев необходима связь «многие-ко-многим». Но эта связь не соответствует правилам нормализации сущностей. На практике выделяют одного из группы составителей.

Атрибутами тестовой базы данных по дисциплине являются имя файла (*.txt), название теста, информация об авторе, количество заданий в тесте (таблица 2.1).

Таблица 2.1 – Структура таблицы «Дисциплина»

№ п/п	Название поля	Обозначение	Тип	Примечание
1	Имя файла базы данных	Filename	строковый	Не более 8 символов, только цифры и буквы латинского алфавита
2	Название теста	Testname	строковый	
3	Автор	Author	строковый	
4	Количество заданий	Kolvozad	целое положительное число	

При реализации было решено оставить её полем в базе данных «Дисциплина», так как информацию будет храниться в текстовом файле, поэтому тип данной связи большой роли не играет. Введение промежуточных сущностей не требуется.

Также было принято решение избавиться от сущности «Студент», так как время, затрачиваемое на ввод студентом данных или регистрацию всех студентов, слишком велико, особенно при небольшом количестве вопросов

тестирования. К тому же присутствующий техперсонал или преподаватель самостоятельно фиксирует результаты студента и может записать помимо результата теста личные данные студента. Автоматическая регистрация результатов тестирования могла бы привести к бесконтрольному списыванию, несамостоятельности работы студента вплоть до ситуации, при которой один студент, выполняя тест, вводит данные другого студента.

Связь между дисциплинами и результатами – «один-ко-многим», так как при тестировании по одному и тому же тесту каждый студент получает свой собственный результат. В то же время результат зависит от конкретной дисциплины.

В университете студенты учатся на разных специальностях. Для каждой специальности существует стандарт, включающий набор дисциплин и перечень тем по ним. Для каждой дисциплины можно составить итоговый тест, состоящий из модулей. Количество модулей может быть любым: от одного (если тест подводит итого по изучению определённой темы или раздела дисциплины), двух, трёх (проведение промежуточных итогов – например, за семестр) до четырёх, пяти или шести (итоговое тестирование по дисциплине).

На втором уровне располагаются учебные модули, содержащие задания по родственным темам. Модуль имеет порядковый номер, название. В каждом модуле определённое количество заданий (таблица 2.2).

Каждый модуль состоит из определённого числа дидактических единиц. Обычно их количество варьируется от 4 до 8. Но если это промежуточное тестирование, то оно может проводиться по одной дидактической единице.

Таблица 2.2 – Структура таблицы «Модуль»

№ п/п	Название поля	Обозначение	Тип
1	Номер модуля	ID	целое положительное число
2	Название модуля	Name	строковый
3	Количество заданий	Kolvozad	целое положительное число

На третьем уровне находятся отдельные дидактические единицы (ДЕ). Они также имеют номер и название темы (таблица 2.3). Каждая дидактическая единицы содержит набор тестовых заданий (общее количество вопросов), из которых выбирается несколько вариантов (количество заданных вопросов).

Таблица 2.3 – Структура таблицы «Дидактическая единица»

№ п/п	Название поля	Обозначение	Тип
1	Номер дидактической единицы	ID	целое положительное число
2	Название дидактической единицы	Tema	строковый
3	Количество вопросов в базе данных	Vvsego	целое положительное число
4	Количество вопросов, которые будут заданы	Vzadan	целое положительное число

Каждая ДЕ содержит как минимум два задания. Обычно их не менее 20. В случае итогового теста по дисциплине из них обычно случайным образом выбирается по одному заданию. Если же тестирование проходит по одной дидактической единице, из базы могут быть отобраны все задания в случайном порядке.

Каждое задание содержит вопрос, правильный ответ и несколько неправильных ответов (таблица 2.4).

Таблица 2.4 – Структура таблицы «Задание»

№ п/п	Название поля	Обозначение	Тип
1	Номер вопроса	ID	целое положительное число
2	Формулировка вопроса	Vopros	строковый
3	Правильный ответ	Answert	строковый
4	Количество неправильных ответов	Otvety	целое положительное число

В соответствии с полем Otvety количество неправильных вариантов может варьироваться от одного до 10-15 и больше. Обычно их три.

При выводе вопроса на экран варианты ответов перемешиваются случайным образом вместе с правильным вариантом ответа.

Количество вариантов ответов можно выделить их в отдельный уровень (таблица 2.5).

Таблица 2.5 – Структура базы данных «Ответ»

№ п/п	Название поля	Обозначение	Тип
1	Номер ответа	ID	целое положительное число
2	Формулировка ответа	Answer	строковый

2.3 Формирование физических таблиц в базе данных

Для реализации базы данных предлагается использовать любой текстовый редактор. Для создания файла необходимо ввести в него структуру вопросов и ответов в соответствии с рисунком 2.5 без элементов форматирования и сохранить в формате *.txt.

В дальнейшем файл можно редактировать. Для форматирования текста можно воспользоваться тегами HTML: использовать жирный шрифт, курсив или подчёркивание, переход на новую строку. В тестах часто используются рисунки (изображение элементов программ, рабочего стола).

Теги HTML также позволяют вставить таблицу либо путём форматирования либо через графический редактор, например Microsoft Paint. Использование тегов форматирования для создания структуры таблицы предпочтительнее. Рисунки необходимо сохранять в формате *.jpg или *.png для корректного отображения на Web-странице.

В физическую структуру базы данных можно бы было добавить ещё ряд сущностей. Ряд дисциплин можно объединить в группы согласно госстандартам или в соответствии с закреплёнными кафедрами. В то же время группы включаются в комплекс дисциплин специальности. Содержание как дисциплины, так и отдельных модулей меняется в зависимости от специальности. Это неизбежно приведёт к появлению связей «многие-ко-многим», которые согласно правилам нормализации требуют исключения путём введения дополнительных сущностей.

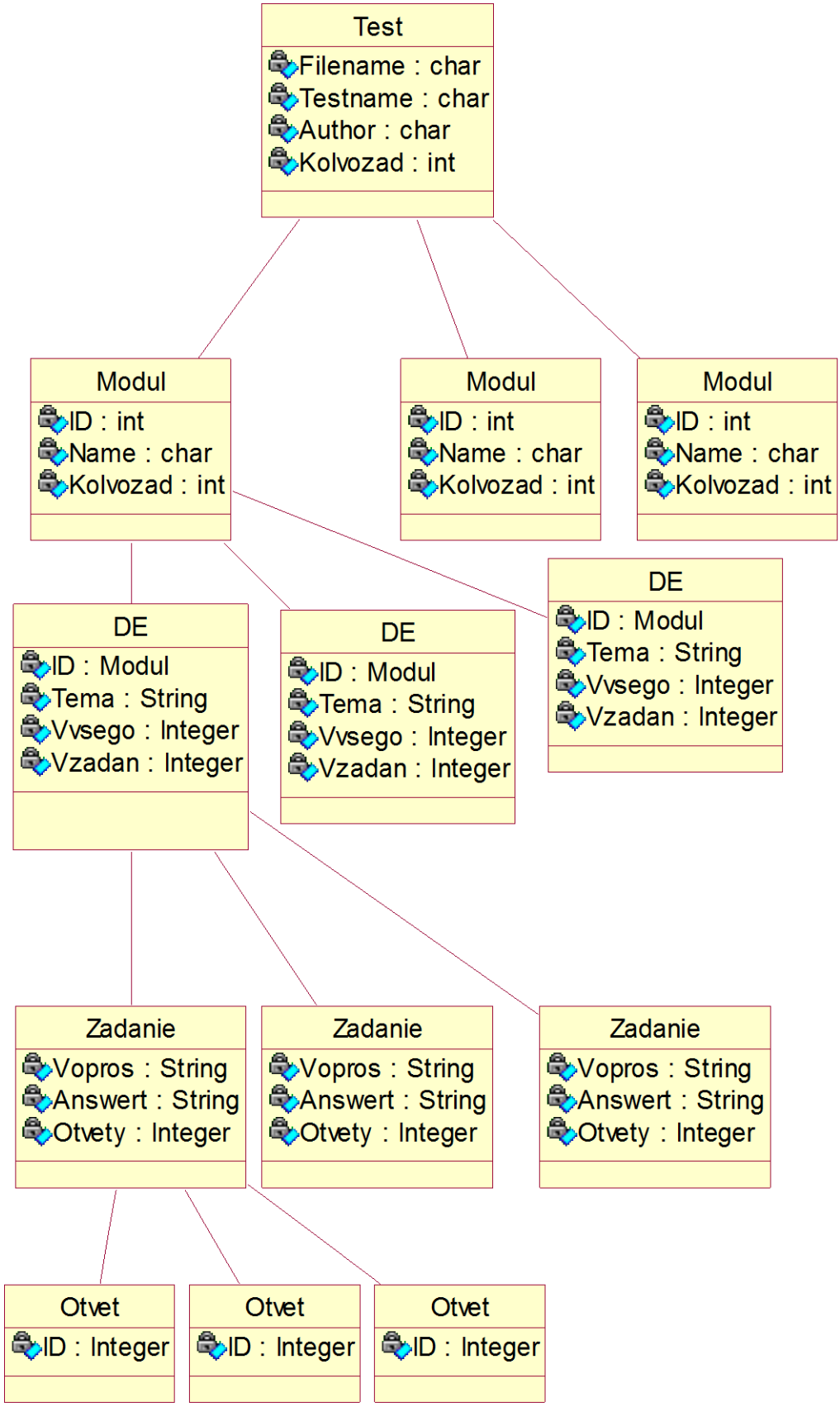

Рисунок 2.5 – Физическая структура базы данных системы онлайн-тестирования

Включение дополнительных сущностей чрезвычайно осложнит структуру базы данных. Создание столь громоздких структур не требуется для успешной реализации заявленной базы и выходит за рамки представленной работы.

Для создания базы данных вполне подойдёт стандартный текстовый редактор для операционных систем семейства Windows Блокнот (Notepad). Блокнот использует оконный класс EDIT.

Вплоть до вышедшей в 2000 году Windows Me текстовый редактор Блокнот поддерживал только самые базовые функции. Многие функции были доступны только из меню, и максимальный размер файла составлял 64 килобайт (предел класса EDIT). В настоящее время редактор поддерживает контекстную замену, горячие клавиши (например, Ctrl-S для сохранения файла), снят предел в 64 Кбайт и добавлена поддержка Юникода.

Кроме Windows, Блокнот способен выполняться также в ReactOS и Wine.

Блокнот автоматически добавляет расширение *.txt, не добавляет к файлу дополнительных элементов форматирования (как это делает Word), быстро работает с файлами, поддерживает большие объёмы данных.

Осталось определиться со структурой базы данных. Первая строчка текстового файла должна содержать информацию об авторе теста – фамилию, инициалы, место работы. Вторая строчка информирует о названии теста. Потом необходимо пропустить одну строчку. Затем начинается сам тест.

Сначала идёт первая группа вопросов (первый модуль, первая дидактическая единица). Тут нужно указать количество вопросов, которые будут задаваться из первой группы. Для указания начала группы используется символ открывающейся скобки «(». Таким образом, четвёртая строчка содержит скобку и количество вопросов, например «(15».

Далее перечисляются все вопросы этой группы. Каждый вопрос начинается с вопросительного знака, т.е. пятая строчка содержит знак «?». Следующая строчка после вопросительного знака (для первого вопроса в базе данных – шестая) содержит сам вопрос. Потом перечисляются варианты ответов. Каждый вариант начинается с восклицательного знака, т.е. седьмая

строка содержит только знак «!».

Первый вариант должен содержать правильный ответ. В случае первого вопроса базы данных верный ответ содержится в восьмой строке. Девятая строка также содержит знак «!». Соответственно десятая строка содержит вариант неправильного ответа. Далее в базе может быть любое количество вариантов ответа для первого вопроса. Отделяются они друг от друга строчками, содержащими восклицательный знак.

После последнего варианта ответа для первого вопроса на следующей строке ставится знак вопроса. Затем на отдельной строке формулируется вопрос аналогично первому. После всех вопросов группы на отдельной строчке ставится закрывающая скобка «)». Потом аналогично первой открывается вторая группа и так далее. После закрытия скобкой последней группы на следующей строчке ставится закрывающая квадратная скобка «]».

После генерации файлов *.php и *.phtml они будут взаимодействовать с собой согласно схеме на рисунке 2.6.

2.4 Проектирование графического пользовательского интерфейса системы онлайн-тестирования

Графический интерфейс пользователя (ГИП, англ. graphical user interface, GUI) в вычислительной технике – система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана (окон, значков, меню, кнопок, списков и т.п.). При этом в отличие от интерфейса командной строки, пользователь имеет произвольный доступ (с помощью клавиатуры или устройства координатного ввода типа «мышь») ко всем видимым экранным объектам. В настоящее время ГИП является стандартной составляющей большинства доступных на рынке операционных систем и приложений.

Интерфейс среды, в которой происходит тестирование, представляет собой Web-интерфейс. По ссылке пользователь попадает на сайт, содержащий названия тестов. Был выбран стандартный тип интерфейса с типичным окном

Web-обозревателя, белым фоном, текстом чёрного цвета, ссылками синего цвета с подчёркиванием, обозначением посещённых ссылок фиолетовым цветом.

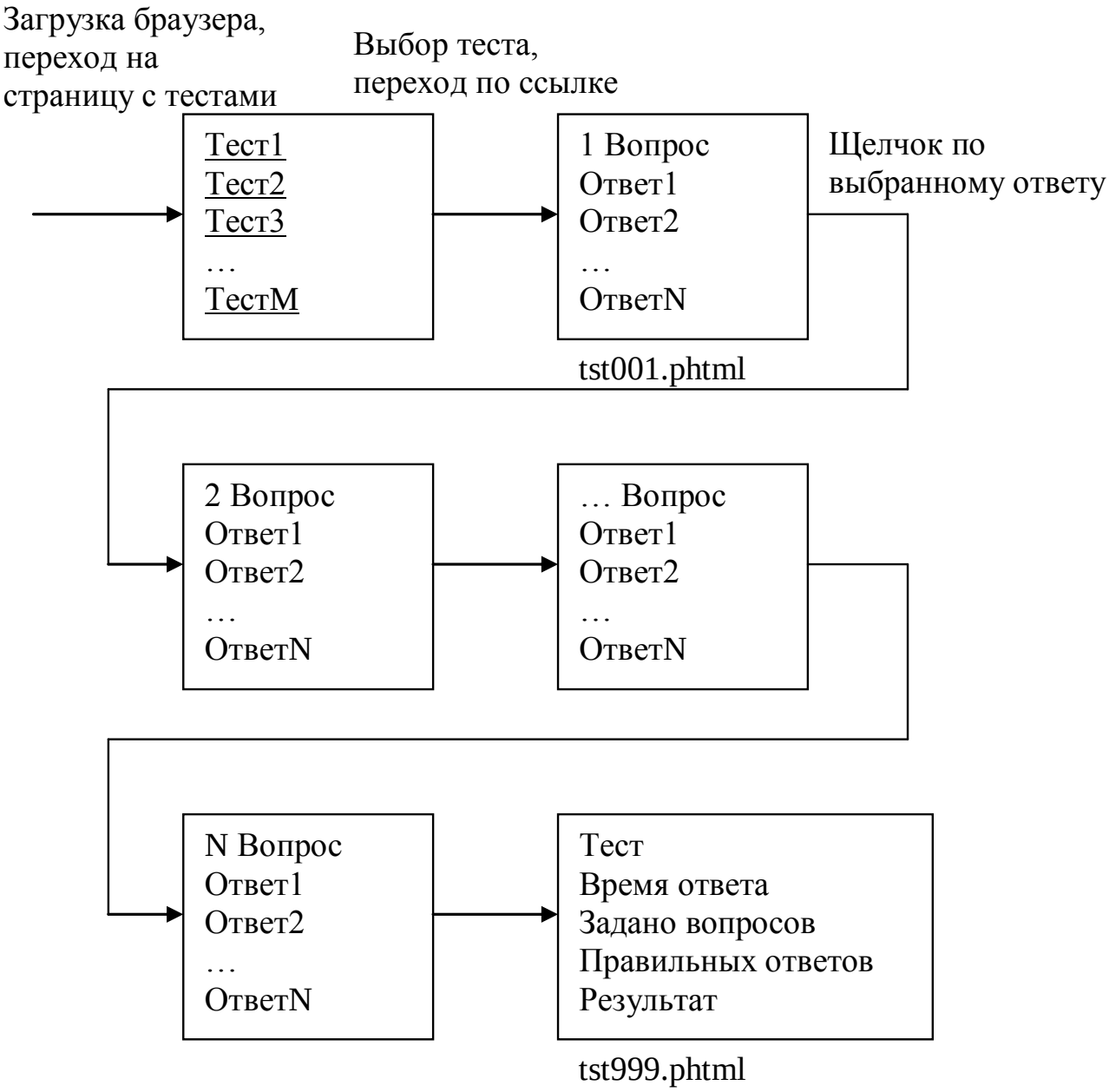

Рисунок 2.6 – Схема взаимодействия web-страниц

Все файлы страниц хранятся на сервере. Для доступа к содержанию сервера необходима программа-клиент. В роли такой программы в рассматриваемом случае выступает браузер. Это может быть Internet Explorer, Mozilla Firefox, Opera и другие клиентские программы.

При щелчке левой кнопкой мыши по ссылке на желаемый тест, пользователю открывается файл tst001.phtml. В заголовке страницы значатся название теста и используемая программа-браузер. Вверху страницы указаны шрифтом красного цвета номер вопроса и на этой же строчке сам вопрос синим цветом. Ниже на отдельных строчках чёрным шрифтом перечислены варианты ответа с указанием порядкового номера ответа. При наведении мыши вариант ответа подсвечивается серым фоном. Для выбора ответа необходимо просто щёлкнуть по нему левой кнопкой мыши. При этом осуществляется переход к следующему вопросу (рисунок 2.7). Не требуется нажатия никаких дополнительных кнопок (например, «Далее»).

При выборе ответа на последний вопрос теста открывается страница с результатом. На первой строчке зелёным шрифтом указано название теста. На второй с точностью до секунды отображается время ответа на тест. На третьей строке указывается количество заданных вопросов. На четвёртой строчке пишется количество правильных ответов. Строки со второй по четвёртую изображены шрифтом чёрного цвета. На пятой строке прописными буквами пишется результат теста. Он зависит от критериев, заданных при формировании теста, и количестве правильных ответов.

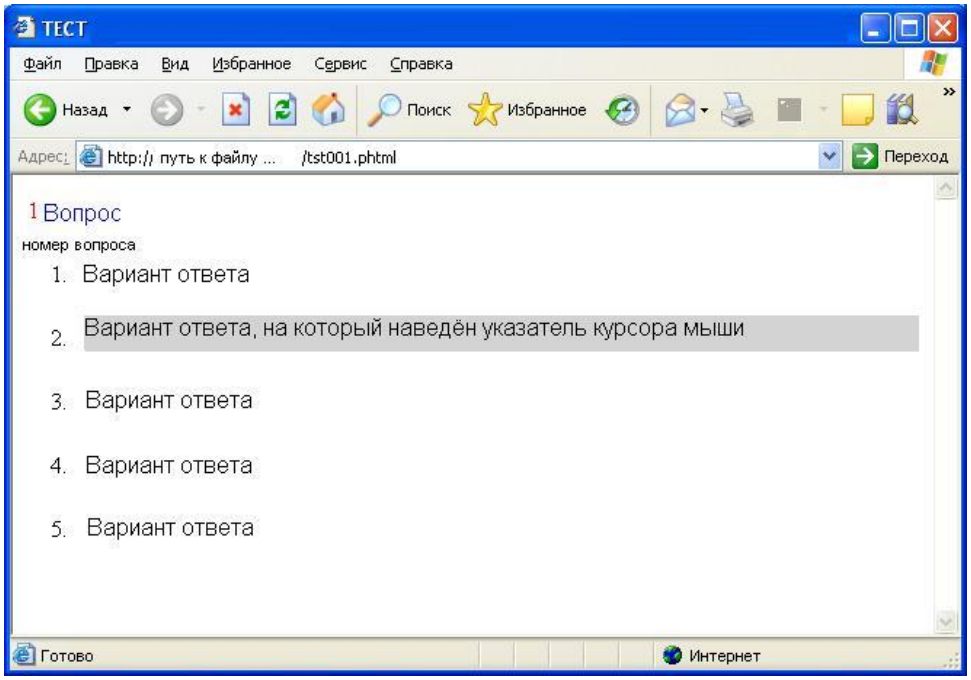

Рисунок 2.7 – Шаблон страницы теста

Оценки «Не зачтено» и «Плохо» пишутся чёрными буквами. Оценка «Удовлетворительно» записывается буквами зелёного цвета. Надпись «Хорошо» указывается синим цветом. Оценки «Зачтено» и «Отлично» выделяются красным цветом (рисунок 2.8).

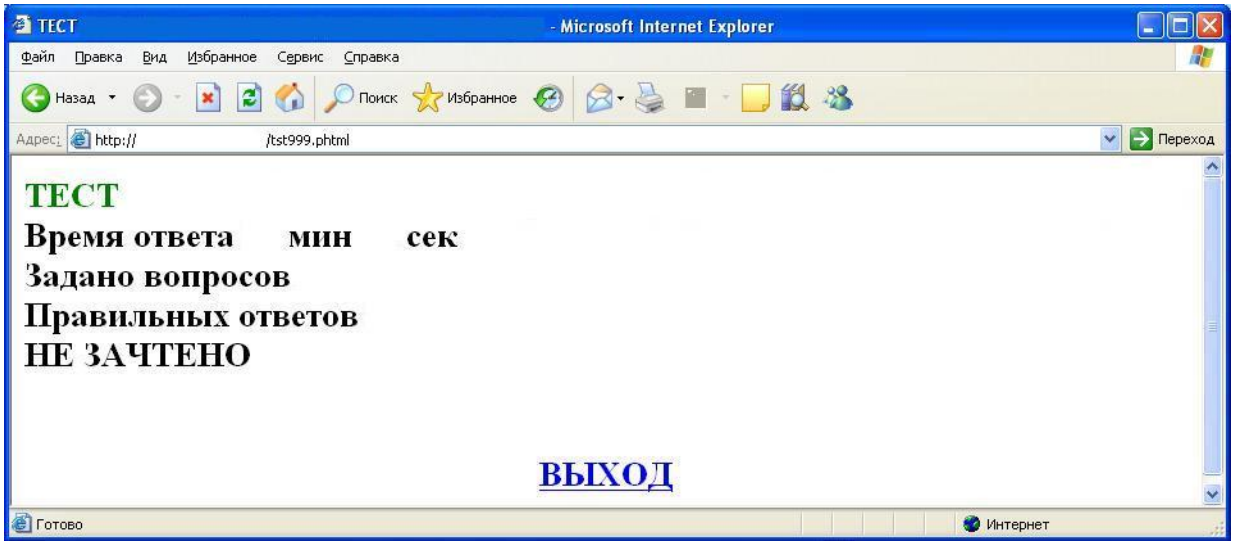

Рисунок 2.8 – Шаблон страницы с результатами теста

Программа для создания тестов содержит один файл lab.exe. При его запуске открывается обычное окно приложения системы Windows. С заголовком синего цвета, строкой меню светло-серого цвета и рабочим окном приложения белого цвета. В строке заголовка отображаются название программы (testMaker) и стандартные кнопки «Свернуть», «Развернуть и «Закрыть». Ниже предложено меню пользователя. Для выхода из программы можно выбрать команду Выход из меню Файл. Для генерации теста в формате PHP 4 необходимо выбрать команду PHP 4 из меню Тесты. При этом откроется окно «Загрузка базы данных», аналогичное стандартному окну открытия файлов. В этом окне отображаются только файлы формата «База данных», то есть файлы с расширением *.txt.

Глава 3 Реализация генератора php-страниц для проведения онлайн-тестирования

3.1 Реализация среды тестирования

Для реализации системы тестирования необходимо установить веб-сервер. Веб-сервер – это программное обеспечение, осуществляющее взаимодействие по HTTP протоколу с браузерами: прием запросов, поиск указанных файлов и передача их содержимого, запуск приложений и передача клиенту результатов их выполнения. Веб-сервер Apache существует как для операционной системы Linux, так и для Windows.

Для сервера используют одну из версий операционной системы Linux. Но для апробации системы тестирования можно воспользоваться web-сервером на базе операционной системы Windows. Воспользуемся версией 1.3.12.

Для работы системы необходимо установить на компьютер, который будет являться сервером, Apache и поставить PHP. Затем необходимо настроить как Apache, так и PHP.

После этого для каждого теста необходимо создать свой каталог на сервере. В корневом каталоге web-пространства был создан файл index.html. В этом файле необходимо прописать ссылки на страницы с тестами (на первую страницу tst001.phtml). Каждый тест должен находиться в отдельном подкаталоге.

Например, если Apache.exe находится по адресу C:\www\apache, а файл index.html – в каталоге C:\www\www1, то файл с тестом можно поместить в директорию C:\www\www1\bd и в файле index.html указать ссылку на файл tst001.phtml этого каталога.

Настроенный Apache и PHP вместе с каталогами тестов можно перенести на другой компьютер путём обычного копирования.

При этом для работы тестов нужно запустить файл apache.exe из каталога C:\www\apache. Затем, не закрывая окно Apache, загрузить окно web-браузера. В адресной строке необходимо указать http://localhost/. После нажатия кнопки «Переход» должна открыться стартовая страница (index.html). Далее нужно

щёлкнуть по ссылке на названии необходимого теста. После ответа на все вопросы выведется страница с результатами. По окончании тестирования можно щёлкнуть по слову «Выход» и закрыть окно Apache простым нажатием на крестик в правом верхнем углу.

3.2 Реализация генератора тестов

При реализации генератора тестов на C++ Builder 10 создаётся стандартное окно. В заголовок окна было добавлено название программы «testMaker». Без изменения оставлены стандартные кнопки «Свернуть», «Развернуть», «Закрыть». Затем было создано главное меню программы. Одним из стандартных решений является наличие пункта меню «Файл». Было принято решение создать подпунктом меню команду «Выход», роль которой аналогична кнопке «Закрыть» и нажатию клавиш [Alt]+[F4]. Предметом разработки будет являться команда «PHP 4», являющаяся командой пункта меню «Тесты».

При вызове этой команды должно открываться диалоговое окно для открытия файлов, которое будет называться «Загрузка базы данных». В этом окне должна отображаться структура папок и файлы с расширением «*.txt» (тип файла «База данных»). В этих файлах заложена структура будущих тестов согласно правилам оформления базы данных.

После открытия файла в рабочем окне генератора тестов анализируется текст базы данных. Также происходит проверка на наличие ошибок. В тексте базы не должно встречаться вложенных групп и количество открывающихся круглых скобок должно соответствовать количеству закрывающихся. Так, например, переменная k изначально равна нулю. Если первым символом строки является открывающаяся скобка «(», то эта переменная инкрементируется (k++). После того, как встретится закрывающаяся скобка «)», переменная k декрементируется (k--). Если в тексте файла встречаются две открывающих скобки подряд (первая скобка не закрывается), программа выдаёт ошибку с указанием номера строки. Наличие ошибки фиксируется с помощью переменной k, которая в этом случае принимает значение 2. В рабочем окне появляется строчка «Открывается вложенная группа».

Если же в тексте файла обнаруживается лишняя закрывающая скобка (открывающаяся скобка не открывалась или была закрыта), переменная k принимает значение -1, на экран выводится номер строки, содержащей ошибку, и формируется сообщение «Закрывается несуществующая группа».

Если при достижении конца файла переменная k равна единице (k==1), выводится сообщение «Не закрыта последняя группа». Также в программе подсчитывается количество вопросов с помощью переменной count. После анализа файла выдаётся результат «Задаётся … вопросов».

Затем открывается диалоговое окно для сохранения файла «Оценки проведённого тестирования» (Form4). Здесь пользователю предлагается определиться с критериями оценок. С помощью двух радиокнопок он может выбрать вид оценки (зачёт или дифференцированная оценка). По умолчанию выбран зачёт. Если выбран «зачёт», пользователь указывает минимальное количество правильных ответов для выставления оценки «Зачтено». По умолчанию стоит «0». Максимальное количество правильных ответов соответствует количеству заданных вопросов.

Если выбрана «дифф. оценка», то активными становятся ячейки для ввода минимального количества правильных ответов на «отлично», «хорошо» и «удовлетворительно». По умолчанию во всех ячейках стоят нули (Edit4->Text="0"; Edit5->Text="0";). Максимальное количество для оценки «отлично» также соответствует количеству вопросов. Максимальное количество верных ответов для оценки «хорошо» равно минимальному значению для оценки «отлично» минус один, а максимум правильных ответов для оценки «удовлетворительно» равен минимальному значению для «хорошо» минус один. До ввода значений максимумы также равны нулю (Label10->Caption="до 0";).

После задания критериев оценки происходит генерация файла tst001.phtml. При этом запускается таймер, происходит запись параметров теста (в том числе ip-адрес тестируемого) в папку /tmp/. Здесь же задаётся кодировка

файла, в информацию файла вставляются данные об авторе, задаются параметры шрифтов вопроса и вариантов ответа.

Аналогичным образом генерируются файлы *.php, *.phtml. Случайным образом задаётся определённое оператором количество вопросов из группы. Случайным образом перемешиваются ответы.

Последнему файлу присваивается имя tst999.phtml. На экран выводится разница между текущим временем и временем начала теста (время ответа). Выводятся переменные, содержащие количество заданных вопросов и количество правильных ответов. В зависимости от выбранного вида оценки и количества правильных ответов красным цветом выводятся надписи «ЗАЧТЕНО» или «ОТЛИЧНО», синим цветом – «ХОРОШО», зелёным цветом – «УДОВЛЕТВОРИТЕЛЬНО», чёрным – «НЕ ЗАЧТЕНО» или «ПЛОХО». Ниже выводится ссылка «ВЫХОД», при нажатии на которую закрывается окно (

 ВЫХОД).

Таким образом, после выбора файла и загрузки базы данных содержимое файла отображается в рабочем окне программы чёрным цветом. По окончании вывода содержимого файла в окне дописывается строка «Задаётся … вопросов», в которой указывается точное количество обнаруженных в базе вопросов. Затем появляется окно «Оценки проведённого тестирования». В нём предлагается определиться с критериями оценок: определить вид оценки (зачёт или дифференцированная оценка) и количество баллов, которое необходимо набрать для получения оценок «Зачтено», «Отлично, «Хорошо» и «Удовлетворительно».

После нажатия кнопки «ОК» открывается окно «Каталог для результата». По умолчанию открывается каталог, из которого была загружена база данных. Можно выбрать каталог, где находятся файлы сервера, и создать отдельную папку для этого теста. При нажатии на кнопку «Сохранить», в этой папке появятся файлы *.php и *.phtml.

3.3 Тестирование системы онлайн-тестирования

Тестирование – это процесс исполнения программы с целью обнаружения ошибки. Тестовый прогон называется удачным, если в результате его обнаружена ошибка, и неудачным, если получен корректный результат.

Практически можно подойти к процедуре написания тестов, используя два подхода:

- стратегия «чёрного ящика»;
- стратегия «белого ящика».

В первом случае тестирование осуществляется с управлением по данным или по входу-выходу. Такое тестирование имеет целью выяснить обстоятельства, в которых поведение программы не соответствует ее внешним спецификациям.

Стратегия «чёрного ящика» использует метод анализа граничных значений. Граничные условия – это ситуации, возникающие непосредственно на границах, выше или ниже границ входных классов эквивалентности. Если входное условие описывает область значений, то строятся тесты для границ области и тесты с неправильными входными данными для ситуации незначительного выхода за границы области. Это правило необходимо исполнить для каждого входного значения.

Данная стратегия подходит для испытания программного модуля тестирования студентов по базе предварительно сформированных вопросов, входящего в состав генератора php-страниц. В зависимости от выбранных вариантов ответа данный тест заканчивается либо удачно (результат тестов привёл к ошибке в работе модуля), либо неудачно (набор предложенных значений корректно распознан и обработан).

Алгоритм тестирования выбранного модуля заключается в следующем. Пользователи, расположенные в локальной вычислительной сети образовательного учреждения, отправляют серверу запросы и получают от него ответ в виде запрашиваемых phtml-файлов. Так как для отправки запроса необходимо задать строку адреса в браузере, то возникает вероятность задания

некорректного адреса или некорректного ввода ссылки в html-файл оператором. Адресная строка должна содержать имя протокола (в данном случае http), имя сервера и относительный адрес файла, поэтому для проверки правильности определения адреса, метод тестирования граничных значений не подходит.

Тестирование по стратегии «белого ящика» подразумевает исследование внутренней структуры текстовой базы данных, входящей в состав системы онлайн-тестирования и осуществляется методом покрытия операторов. Критерием покрытия операторов является прохождение каждого теста хотя бы десять раз. Ошибки обычно проявляются в виде некорректных ответов (два правильных ответа) и в неправильных ссылках на картинки (отсутствие файла с картинкой или неверное написание).

Реализация корректной обработки запросов в системе онлайн-тестирования полностью обеспечивается веб-сервером. Это позволяет избежать ошибок при построении запроса и гарантирует отсутствие выдачи ложных результатов пользователю (студенту). Целостность данных также обеспечивается внутренними механизмами веб-сервера.

Заключение

Целью бакалаврской работы была разработка генератора php-страниц для проведения онлайн-тестирования.

В работе были проанализированы современные методы тестирования. В качестве средства структурного анализа было выбрано CASE-средство Rational Rose. Для построения информационной модели и структуры базы данных было выбрано приложение Design/IDEF (методологии IDEF0 и IDEF1X).

В качестве программного обеспечения был выбран веб-интерфейс из соображений удобства работы с ним, особенно в клиент-серверной среде. В дальнейшем планируется организовать единое образовательное пространство университета для полноценного доступа ко всем имеющимся электронным ресурсам. При проектировании компонентов веб-интерфейса были использованы текстовый редактор Блокнот для написания исходного кода стартовой страницы и язык PHP.

Для реализации генератора тестов была выбрана система программирования C++ Builder 10. Для предоставления тестового материала в наглядной форме были использованы все возможности языка HTML (вставка рисунков, таблиц, форматирование текста).

Созданный программный комплекс содержит средства контроля знаний, выполненные в форме теста. Путём тестирования проверяются также полученные обучаемыми практические навыки. Выполняя тест, студент проверяет свои теоретические знания, почерпнутые из цикла обучения.

Таким образом, все поставленные задачи были выполнены и цель работы достигнута.

Главной функцией разработанного генератора php-страниц для проведения онлайн-тестирования является возможность проверки знаний по дисциплине без преподавателя, формирование привычки обучаемого к самообразованию.

Список используемой литературы

Монографии

1. Громцев С.А. Педагогические проблемы системы подготовки специалистов с высшим образованием в Российской Федерации: монография/ Громцев С.А., Пальчиков А.Н., Коновалов В.Б. – Саратов: Вузовское образование, 2014. – 65 с.

Учебники и учебные пособия

2. Бенкен Е.С. PHP, MySQL, XML: программирование для Интернета. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2011. – 287 с.

3. Головина Е.Ю. Интеллектуальные методы для создания систем поддержки принятия решений: учебное пособие/ Головина Е.Ю. – М.: Издательский дом МЭИ, 2011. – 104 с.

4. Дронов В.А. HTML 5, CSS 3 и Web 2.0. Разработка современных Web-сайтов. – СПб.: БХВ-Петербург, 2011. – 414 с.

5. Колисниченко Д.Н. PHP 5/6 и MySQL 6. Разработка Web-приложений. – 3-е изд., перераб. и доп. – СПб.: БХВ-Петербург, 2011. – 520 с.

6. Кузнецов М.В. PHP на примерах : [для программистов и Web-разработчиков] / М.В. Кузнецов, И.В. Симдянов. – СПб: Питер, 2012. – 398 с.

7. Ллойд Й. Создай свой веб-сайт с помощью HTML и CSS = Build Your Own Website The Right Way Using HTML & CSS / Й. Ллойд ; [пер. с англ. О. Потапова]. – СПб: Питер, 2013. – 412 с.

8. Немцова Т.И. Практикум по информатике. Ч. 2. Компьют. графика и Web-дизайн. Практ.: Уч. пос. / Т.И. Немцова и др.; Под ред. Л.Г. Гагариной – М.: ИД ФОРУМ: ИНФРА-М, 2013. – 288 с.

9. Никсон Р. Создаём динамические веб-сайты с помощью PHP, MySQL, JavaScript и CSS = Learning PHP, MySQL, JavaScript and CSS / Р. Никсон ; [пер. с англ. Н. Вильчинский]. – 2-е изд. – СПб: Питер, 2013. – 560 с.

10. Основы информационных технологий / С.В. Назаров [и др.]. – М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016. – 530 с.

11. Пожарина Г.Ю. Стратегия внедрения свободного программного обеспечения в учреждениях образования / Пожарина Г.Ю., Поносов А.М. – М.: БИНОМ. Лаборатория знаний, 2012. – 153 с.

12. Самуйлов С.В. Объектно-ориентированное моделирование на основе UML: учебное пособие/ Самуйлов С.В. – Саратов: Вузовское образование, 2016. – 37 с.

13. Смирнов А.А. Технологии программирования: учебное пособие/ Смирнов А.А., Хрипков Д.В. – М.: Евразийский открытый институт, 2011. – 191 с.

14. Тузовский А.Ф. Проектирование и разработка web-приложений : учеб. пособие / А.Ф. Тузовский. – Томск: ТПУ, 2014. – 218 с.

15. Фатеев А.М. Информационные технологии в педагогике и образовании: учебное пособие для студентов-бакалавров по направлениям 050100 – «Педагогическое образование» и 050400 – «Психолого-педагогическое образование»/ Фатеев А.М. – М.: Московский городской педагогический университет, 2012. – 200 с.

Литература на иностранном языке

16. Johanan Joshua, Khan Talha, Zea Ricardo. Web Developer's Reference Guide. – Packt Publishing, 2016. – 838 p.

17. Lloyd I. Build Your Own Website The Right Way Using HTML & CSS. – 3rd edition, SitePoint, Collingwood, Australia, 2011. – 514 p.

18. MacDonald M. Creating a Website: The Missing Manual. – 4th Edition, O'Reilly, 2015. – 624 p.

19. Nacke K. D Web Development. – Packt Publishing, 2016. – 196 p.

20. Nixon R. Learning PHP, MySQL, JavaScript and CSS. – 2nd Edition, O'Reilly, 2012. – 582 p.

21. Nixon R. Web Developer's Cookbook. – McGraw-Hill, 2012. – 992 p.

22. Pollock P. Web Hosting For Dummies. – John Wiley & Sons, Inc., 2013. – 360 p.

23. Sauble Daniel. Offline First Web Development: Design and build robust offline-first apps for exceptional user experience even when an internet connection is absent. – Packt Publishing, 2015. – 316 p.

24. Tim Ash, Maura Ginty, Rich Page. Landing Page Optimization. – Sybex, 2012. – 496 p.

25. Uhlmann T. Instant Lift Web Applications. – Packt Publishing, 2013. – 336 p.

26. Wellens P. Practical Web Development. – Packt Publishing, 2015. – 276 p.

Приложение

Листинг файла lab.c

```
void __fastcall TForm1::HTML1Click(TObject *Sender)
{
  int first=1,n=1,last=0, k, dummy, bold, eold, eold2, qq;
  AnsiString fn;
  char r[20]="001";
  FILE *f2, *f3, *f4, *ftemp;
  char str[250];
  char fname[25]="tst";
  int i,j,begin=1,end=2, ans, que, count;
  Memo1->Clear();
  if(OpenDialog1->Execute()){
//      Memo1->Lines->LoadFromFile(OpenDialog1->FileName);
       sprintf(str,"%s",OpenDialog1->FileName);
       f2=fopen(str,"rt");
       fgets(str,240,f2); str[(i=strlen(str))-1]=0;
       Memo1->Lines->Append(str);
       i=count=k=ans=0;j=1; do{ j++;
       fgets(str,240,f2);
       i=strlen(str);
       if(i==1)continue;
        str[i-1]=0;
       Memo1->Lines->Append(str);
       switch(str[0]){
       case '(':
         k++;  ans=0;
         if(k==2){
           Memo1->Lines->Append("");
           sprintf(str,"В строке %d ошибка",j);
           Memo1->Lines->Append(str);
           Memo1->Lines->Append("Открывается вложенная группа");
           return;
         }
         que=1; sscanf(str+1,"%d",&que);
         break;
       case ')':
         k--;
         if(que>ans)que=ans;
         if(que<2)count++; else count+=que;
         if(k==-1){
           Memo1->Lines->Append("");
           sprintf(str,"В строке %d ошибка",j);
           Memo1->Lines->Append(str);
           Memo1->Lines->Append("Закрывается несуществующая группа");
           return;
         } break;
       case '?':
         if(k==0)count++;
         ans++;
         break;
```

```
        }  }while(!feof(f2)); fclose(f2);

        if(k==1) Memo1->Lines->Append(")");
        Memo1->Lines->Append("]");
           Memo1->Lines->Append("");
           sprintf(str,"Задаётся %d вопросов",count);
    if(k==1)    Memo1->Lines->Append("*** Не закрыта последняя группа ***");
           Memo1->Lines->Append(str);
        mark_ball=count;
        Form4->ShowModal();
    //       return;
    //

    //     Form4->ShowModal();
        SaveDialog3->FileName=fname;
    //     strcpy(fname,"t");
        if(SaveDialog3->Execute()){
        fn=SaveDialog3->FileName;
        sprintf(str,"%s001.phtml",fn);
        f2=fopen(str,"wt");
    for(;;){
            if(first==0)goto NOFIRST;
            fprintf(f2,"%s\n","<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2//EN\"><html><head><title>");
            fprintf(f2,"%s\n",Memo1->Lines->Strings[1]);
            fprintf(f2,"</title><meta http-equiv=\"content-type\" content=\"text/html; charset=Windows-1251\"/>\n");
            fprintf(f2,"<meta name=\"author\" content=\"%s\"/>\n",Memo1->Lines->Strings[0]);
            fprintf(f2,"<meta name=\"Generator\" content=\"Generated with taskMaker by mstrogov@list.ru\"/>\n");
            fprintf(f2,"%s\n","<style type=\"text/css\"> ");
            fprintf(f2,"%s\n",".que {color:blue; font-size=large;}");
            fprintf(f2,"%s\n",".a {  font-family: Verdana, Arial, Helvetica, sans-serif;");
            fprintf(f2,"%s\n","  font-size: 15pt; color: black;  text-decoration: none }");
    //        fprintf(f2,"%s\n","a:hover { font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;");
    //        fprintf(f2,"%s\n"," font-size: 15pt; font-style: normal; color: #FF1538; text-decoration: underline }");
            fprintf(f2,"%s\n","</style> ");
            fprintf(f2,"%s\n"," <?  $Q=\"\"; $PARM=\"\"; $ANS=2; $TIME=time();");
            fprintf(f2,"  $IS=(time()%%100000)*1000+(time()/100000)%%1000; mt_srand($IS);  $R="__HOST__";  $f=fopen(\"/tmp/\".$R,\"w\");  fputs($f,\"0:0:$IS:$TIME\");  \n ?>");
            fprintf(f2,"%s\n","<script language=\"JavaScript\">");
            fprintf(f2,"%s\n","  function ans(a){");
            fprintf(f2,"%s\n","  document.params.ANS.value=a;");
    //        fprintf(f2,"%s\n","  event.cancelBubble=true;");
            fprintf(f2,"%s\n","  document.params.exit.click(); }");
            fprintf(f2,"%s\n","</script> ");

    NOFIRST:
```

```
            qq=eold=end;
            getQ(end-1,&begin,&end,&ans,&que,&last, &count);
        sprintf(str,"%s%03d.phtml",fn,last?999:n+1);
        f3=fopen(str,"wt");
            if(que<2)que=0;
            if(count<2)count=0;
            if(count>que) count=que;

    if(count>0){ sprintf(str,"%s%03d.php",fn,n);
            f4=fopen(str,"wt");
            fprintf(f4,"%s\n","<?  $R="__HOST__";  $f=fopen(\"/tmp/\".$R,\"r\");
if($f==0){echo  \"<html><body><span  style='font-size:30pt;'>Server
error</span></body></html>\"; exit();} else{$str=fgets($f,100); fclose($f); $p=explode(\":\",$str);
$QS=$p[0]; $NB=$p[1];  $IS=$p[2]; $TIME=$p[3]; $answer=$p[4];} ?>");
            fprintf(f4,"%s\n","<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML
3.2//EN\"><html><head><title>");
            fprintf(f4,"%s\n",Memo1->Lines->Strings[1]);
            fprintf(f4,"</title><meta  http-equiv=\"content-type\"  content=\"text/html;
charset=Windows-1251\"/>\n");
            fprintf(f4,"<meta  name=\"author\"  content=\"%s\"/>\n",Memo1->Lines-
>Strings[0]);
            fprintf(f4,"<meta name=\"Generator\" content=\"Generated with taskMaker by
mstrogov@list.ru\"/>\n");
    //          fprintf(f4,"%s\n","</title><meta http-equiv=\"content-type\" content=\"text/html;
charset=Windows-1251\">");
            fprintf(f4,"%s\n","<style type=\"text/css\"> ");
            fprintf(f4,"%s\n",".que {color:blue; font-size=large;}");
            fprintf(f4,"%s\n",".a {  font-family: Verdana, Arial, Helvetica, sans-serif;");
            fprintf(f4,"%s\n","  font-size: 15pt; color: black;  text-decoration: none }");
    //          fprintf(f4,"%s\n","a:hover { font-family: Verdana, Geneva, Arial, Helvetica, sans-
serif;");
    //          fprintf(f4,"%s\n"," font-size: 15pt; font-style: normal; color: #FF1538; text-
decoration: underline }");
            fprintf(f4,"%s\n","</style> ");
            fprintf(f4,"%s\n","<script language=\"JavaScript\">");
            fprintf(f4,"%s\n","  function ans(a){");
            fprintf(f4,"%s\n","  document.params.ANS.value=a;");
      //          fprintf(f4,"%s\n","  event.cancelBubble=true;");
            fprintf(f4,"%s\n","  document.params.exit.click(); }");
            fprintf(f4,"%s\n","</script> ");

    } //if(count>0)
    //          sprintf(str,"%s%03d.phtml",fn,(last==1)?999:n+1);
            fprintf(f3,"%s\n","<?  $R="__HOST__";  $f=fopen(\"/tmp/\".$R,\"r\");
if($f==0){echo  \"<html><body><span  style='font-size:30pt;'>Server
error</span></body></html>\"; exit();} else{$str=fgets($f,100); fclose($f); $p=explode(\":\",$str);
$QS=$p[0]; $NB=$p[1];  $IS=$p[2]; $TIME=$p[3]; $answer=$p[4];} ?>");
            fprintf(f3,"%s\n","<!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML
3.2//EN\"><html><head><title>");
            fprintf(f3,"%s\n",Memo1->Lines->Strings[1]);
```

```
        //              fprintf(f3,"%s\n","</title><meta http-equiv=\"content-type\" content=\"text/html; charset=Windows-1251\">");
                        fprintf(f3,"</title><meta http-equiv=\"content-type\" content=\"text/html; charset=Windows-1251\"/>\n");
                        fprintf(f3,"<meta name=\"author\" content=\"%s\"/>\n",Memo1->Lines->Strings[0]);
                        fprintf(f3,"<meta name=\"Generator\" content=\"Generated with taskMaker by mstrogov@list.ru\"/>\n");
                        fprintf(f3,"%s\n","<style type=\"text/css\"> ");
                        fprintf(f3,"%s\n",".que {color:blue; font-size=large;}");
        if(last==0){
                        fprintf(f3,"%s\n",".a {         font-family: Verdana, Arial, Helvetica, sans-serif;");
                        fprintf(f3,"%s\n","    font-size: 15pt; color: black;  text-decoration: none }");
        //              fprintf(f3,"%s\n","a:hover { font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;");
        //              fprintf(f3,"%s\n"," font-size: 15pt; font-style: normal; color: #FF1538; text-decoration: underline }");
        }
                        fprintf(f3,"%s\n","</style> ");
        if(last==0){
                        fprintf(f3,"%s\n","<script language=\"JavaScript\">");
                        fprintf(f3,"%s\n","     function ans(a){");
                        fprintf(f3,"%s\n","     document.params.ANS.value=a;");
         //             fprintf(f3,"%s\n","     event.cancelBuble=true;");
                        fprintf(f3,"%s\n","     document.params.exit.click(); }");
                        fprintf(f3,"%s\n","</script> ");
        }
                        sprintf(r,"%03d",n);
                        fprintf(f2,"%s\n","   </head><body><?");
        if(count>0)  fprintf(f4,"%s\n","   </head><body>");
                        sprintf(str,"<? mt_srand($IS);");
                        fprintf(f3,"%s\n",str);
        if(count>0)  fprintf(f4,"%s\n",str);


        if(que>0){ //  группа вопросов
                        sprintf(str,"  $qq=mt_rand(0,%d);",que-1);
                        fprintf(f2,"%s\n",str);
        if(count==0) fprintf(f3,"%s\n",str);
        if(count>0){
        //              fprintf(f4,"%s\n",str);
                        fprintf(f2,"%s\n"," $PARM=sprintf(\"%02d\",$qq); mt_srand($IS); ");
                        fprintf(f3,"%s\n"," $qq=(int)$PARM;");
        //              fprintf(f4,"%s\n"," $qq=(int)substr($PARM,0,2);");
        } // if(count>0)
                        fprintf(f2,"%s\n","switch($qq){");
        //              fprintf(f3,"%s\n","switch($qq){");
        //if(count>0)  fprintf(f4,"%s\n","switch($qq){");
        for(k=0;k<que;k++){ // каждый вопрос группы
                        sprintf(str,"case %d:",k);
                        fprintf(f2,"%s\n",str);
        //              fprintf(f3,"%s\n",str);
```

```
        //if(count>0)  fprintf(f4,"%s\n",str);
                fprintf(f2,"%s\n","echo \"<span class=\\\"que\\\">\";");
                for (i=begin;i<end;i++){ // печать вопроса
                  sprintf(str,"echo \" %s\\n\";",Memo1->Lines->Strings[i]);
                  fprintf(f2,"%s\n",str);
                } //for (i=begin;i<end;i++)
                fprintf(f2,"%s\n","echo \"</span><ol>\";");
                sprintf(str,"  unset($g);$g=array();  $ans=mt_rand(0,%d);  for ($ii=0;$ii<%d;$ii++)$g[]=$ii; for ($ii=0;$ii<300;$ii++)",ans-1,ans);
                fprintf(f2,"%s\n",str);
        //        fprintf(f3,"%s\n",str);
        //if(count>0)  fprintf(f4,"%s\n",str);
                sprintf(str,"{  $t=$g[$m=mt_rand(1,%d)];  $g[$m]=$g[$n=mt_rand(1,%d)]; $g[$n]=$t;} $g[0]=$g[$ans]; $g[$ans]=0; fputs($f,\":$ans\"); fclose($f);",ans-1,ans-1);
                fprintf(f2,"%s\n",str);
        //        fprintf(f3,"%s\n",str);
        //if(count>0)  fprintf(f4,"%s\n",str);
                sprintf(str,"for($i=0;$i<%d;$i++){",ans);
                fprintf(f2,"%s\n",str);
                fprintf(f2,"%s\n"," switch ($g[$i]){");
        //        fprintf(f3,"%s\n","   $QS++; if($g[$ANS]==0)$NB++; break;");
        //if(count>0)  fprintf(f4,"%s\n","   $QS++; if($g[$ANS]==0)$NB++; break;");

                for (i=0; i<ans;i++){
                  sprintf(str,"case %d:",i);
                  fprintf(f2,"%s\n",str);
                  fprintf(f2,"%s\n","echo  \"<li><div  class='a' onMouseOver='this.style.background=\\\"aqua\\\"' onMouseOut='this.style.background=\\\"white\\\"' onClick='ans(\".$i.\")'>\\n\"; ");
        //          запрашиваем ответ
                  getA(end-1,&begin,&end);
                 // печать ответа
                  for (j=begin;j<end;j++){
                    sprintf(str,"  print \"  %s\\n\";",Memo1->Lines->Strings[j]);
                    fprintf(f2,"%s\n",str);
                  }// for (j=begin;j<end;j++)
                fprintf(f2,"%s\n","  echo \"</div><br/></li>\\n\"; break;");
                }// закончили ответы вопроса
                fprintf(f2,"%s\n"," }  } break;");
                if(k!=que-1)getQ(end-1,&begin,&end,&ans,&dummy,&dummy, &dummy);
        } // все ответы вопроса группы перебраны
                fprintf(f2,"%s\n","}  ?>");
        //}
        if(last!=1)  fprintf(f3,"%s\n","  $R="__HOST__";  $f=fopen(\"/tmp/\".$R,\"w\"); fputs($f,$QS.\":\".$NB.\":\".$IS.\":\".$TIME);  mt_srand($IS); ?>");

        if(count>0){ // несколько вопросов из группы  заполняем php
                fprintf(f4,"%s\n","  if($ANS==$answer)$NB++;  $QS++;  $Q--; $IS=(time()%100000)*1000+(time()/100000)%1000; mt_srand($IS); ");
        if(last==0)
                sprintf(str,"if($Q==0)$NEXT=\"%s%03d.phtml\";  else  $NEXT=\"%s%03d.php\"; $w=array();",fname,n+1,fname,n);
```

```
        else
                sprintf(str,"if($Q==0)$NEXT=\"%s999.phtml\";  else  $NEXT=\"%s%03d.php\";
$w=array();",fname,fname,n);
                fprintf(f4,"%s\n",str);
                fprintf(f4,"%s\n",          "$R="__HOST__";       $f=fopen(\"/tmp/\".$R,\"w\");
fputs($f,$QS.\":\".$NB.\":\".$IS.\":\".$TIME); " );
                fprintf(f4,"%s\n","$k=strlen($PARM);
for($i=0;$i<(int)($k/2);$i++)$w[(int)(substr($PARM,2*$i,2))]=1;");
                eold2=end;
                getQ(qq-1,&begin,&end,&ans,&dummy,&dummy, &dummy);
                sprintf(str,"do      {      $qq=mt_rand(0,%d);}      while      (isset($w[$qq]));
mt_srand($IS);if($Q==0)$PARM=sprintf(\"%%02d\",$qq);                                    else
$PARM=sprintf(\"%%02d\",$qq).$PARM;",que-1,que-1,que-1);
                fprintf(f4,"%s\n",str);
                fprintf(f4,"%s\n"," switch($qq){");
        for(k=0;k<que;k++){ // для каждого возможного вопроса
                sprintf(str,"case %d:",k);
                fprintf(f4,"%s\n",str);
                fprintf(f4,"%s\n","echo \"<span class=\\\"que\\\">\";");
                for (i=begin;i<end;i++){
                  sprintf(str,"echo \" %s\\n\";",Memo1->Lines->Strings[i]);
                  fprintf(f4,"%s\n",str);
                }// for (i=begin;i<end;i++)
                fprintf(f4,"%s\n","echo \"</span><ol>\";");
                sprintf(str,"          unset($g);$g=array();          $ans=mt_rand(0,%d);          for
($ii=0;$ii<%d;$ii++)$g[]=$ii; for ($ii=0;$ii<300;$ii++)",ans-1,ans);
                fprintf(f4,"%s\n",str);
                sprintf(str,"{      $t=$g[$m=mt_rand(1,%d)];      $g[$m]=$g[$n=mt_rand(1,%d)];
$g[$n]=$t;} $g[0]=$g[$ans]; $g[$ans]=0; fputs($f,\":$ans\"); fclose($f);",ans-1,ans-1);
                fprintf(f4,"%s\n",str);
                sprintf(str,"for($i=0;$i<%d; $i++) switch($g[$i]){",ans);
                fprintf(f4,"%s\n",str);
          for (i=0; i<ans;i++){ // каждый ответ

                  sprintf(str,"case %d:",i);
                  fprintf(f4,"%s\n",str);
                  fprintf(f4,"%s\n","echo                  \"<li><div                  class='a'
onMouseOver='this.style.background=\\\"aqua\\\"'
onMouseOut='this.style.background=\\\"white\\\"' onClick='ans(\".$i.\")'>\\n\"; ");
                  getA(end-1,&begin,&end);
                  for (j=begin;j<end;j++){
                    sprintf(str,"  print \"  %s\\n\";",Memo1->Lines->Strings[j]);
                    fprintf(f4,"%s\n",str);
                  } // for (j=begin;j<end;j++)
                fprintf(f4,"%s\n","  echo \"</div><br/></li>\\n\"; break;");
          }  //for (i=0; i<ans;i++)
                fprintf(f4,"%s\n"," }  break;");

                if(k!=que-1)getQ(end-1,&begin,&end,&ans,&dummy,&dummy, &dummy);
                sprintf(str,"     unset($g);$g=array();     for     ($ii=0;$ii<%d;$ii++)$g[]=$ii;     for
($ii=0;$ii<300;$ii++)",ans);
                fprintf(f4,"%s\n","");
```

```
        } // вопросы группы закончились
                fprintf(f4,"%s\n"," } ?></ol>");
                end=eold2;
                fprintf(f4,"%s\n","");
                fprintf(f4,"%s\n","");
        //}//           fprintf(f2,"%s\n","}  ?>");
        } // count > 0
        } // que > 0

        // одиночный вопрос

        else{
        //            fprintf(f3,"%s\n"," }  ");
                fprintf(f2,"%s\n","echo \"<span class=\\\"que\\\">\";");
                for (i=begin;i<end;i++){  // вывод вопроса
                  sprintf(str,"echo \" %s\\n\";",Memo1->Lines->Strings[i]);
                  fprintf(f2,"%s\n",str);
                  }
                sprintf(str,"    echo \"</span><ol>\"; $q=array(); for ($i=0;$i<%d;$i++)$q[]=$i; for
($i=0;$i<300;$i++)",ans);
                fprintf(f2,"%s\n",str);
                sprintf(str,"          $q=array();      for      ($i=0;$i<%d;$i++)$q[]=$i;      for
($i=0;$i<300;$i++)",ans);
                fprintf(f3,"%s\n",str);
                sprintf(str,"{      $t=$q[$m=mt_rand(0,%d)];      $q[$m]=$q[$n=mt_rand(0,%d)];
$q[$n]=$t;} for($i=0;$i<%d;$i++)switch($q[$i]){",ans-1,ans-1,ans);
                fprintf(f2,"%s\n",str);
                sprintf(str,"{      $t=$q[$m=mt_rand(0,%d)];      $q[$m]=$q[$n=mt_rand(0,%d)];
$q[$n]=$t;}",ans-1,ans-1);
                fprintf(f3,"%s\n",str);
                fprintf(f3,"%s\n","   $QS++; if($q[$ANS]==0)$NB++; ");
        if(last==0)     fprintf(f3,"%s\n","unset($q);$R="__HOST__";  $f=fopen(\"/tmp/\".$R,\"w\");
fputs($f,$QS.\":\".$NB.\":\".$IS.\":\".$TIME);   mt_srand($IS); ?>");
                for (i=0; i<ans;i++){
                  sprintf(str,"case %d:",i);
                  fprintf(f2,"%s\n",str);
                fprintf(f2,"%s\\n\n","echo                \"<li><div                class='a'
onMouseOver='this.style.background=\\\"aqua\\\"'
onMouseOut='this.style.background=\\\"white\\\"' onClick='ans(\".$i.\")'>\"; ");
                  getA(end-1,&begin,&end);
                  for (j=begin;j<end;j++){
                    sprintf(str,"  print \"  %s\\n\";",Memo1->Lines->Strings[j]);
                    fprintf(f2,"%s\n",str);
                  }
                fprintf(f2,"%s\n","  echo \"</div><br/></li>\";  break;");
                }  // перебрали ответы  for (i=0; i<ans;i++){
                fprintf(f2,"%s\n","}?></ol>");

        }  //    одиночный ответ

        if(count==0){
```

```
                sprintf(str,"<form name=\"params\" action=\"%s%03d.phtml\" method=\"POST\" style=\"visibility:hidden\">",fname,((last==0)?n+1:999));
                fprintf(f2,"%s\n",str);
                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"Q\" value=\"\"/>");
                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"PARM\" value=\"<?echo $PARM?>\"/>");
        }else{ //count!=0
                sprintf(str,"<form name=\"params\" action=\"%s%03d.php\" method=\"POST\" style=\"visibility:hidden\">",fname,n);
                fprintf(f2,"%s\n",str);
                sprintf(str,"<form name=\"params\" action=\"<?echo $NEXT;?>\" method=\"POST\" style=\"visibility:hidden\">");
                fprintf(f4,"%s\n",str);
                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"PARM\" value=\"<?echo $PARM?>\"/>");
                sprintf(str,"<input type=\"hidden\" name=\"Q\" value=\"%d\"/>",count-1);
                fprintf(f2,"%s\n",str);
                sprintf(str,"<input type=\"hidden\" name=\"Q\" value=\"<?echo $Q;?>\"/>");
                fprintf(f4,"%s\n",str);
                fprintf(f4,"%s\n","<input type=\"hidden\" name=\"PARM\" value=\"<?echo $PARM?>\"/>");
        //                fprintf(f4,"%s\n","<input type=\"hidden\" name=\"NB\" value=\"<?echo $NB?>\"/>");
        //                fprintf(f4,"%s\n","<input type=\"hidden\" name=\"ISK\" value=\"<?echo $IS?>\"/>");
                fprintf(f4,"%s\n","<input name=\"exit\" type=\"Submit\" value=\"exit\"/>");
                fprintf(f4,"%s\n","<input type=\"hidden\" name=\"ANS\" value=\"<?echo $ANS?>\"/>");
        //                fprintf(f4,"%s\n","<input type=\"hidden\" name=\"QS\" value=\"<?echo $QS?>\"/>");
                fprintf(f4,"%s\n","</form></body></html>");
                fprintf(f4,"%s\n","");
                fclose(f4);
        }  //if(count==0)

        //                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"NB\" value=\"<?echo $NB?>\"/>");
                fprintf(f2,"%s\n","<input name=\"exit\" type=\"Submit\" value=\"exit\"/>");
                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"ANS\" value=\"<?echo $ANS?>\"/>");
        //                fprintf(f2,"%s\n","<input type=\"hidden\" name=\"QS\" value=\"<?echo $QS?>\"/>");
                fprintf(f2,"%s\n","</form></body></html>");
                fprintf(f2,"%s\n","");
        //          sprintf(str,"%s%03d.phtml",fn,n+1);
                fclose(f2);
                n++; first=0;
                f2=f3;
                if(last==1)goto LAST;
                continue;
        LAST:
        if(que>0)
```

```
                fprintf(f2,"%s\n","     echo \" </head><body><span style='font-size:20pt; font-weight:700;'><font color=\\\"green\\\">\\n\";");
        else
                fprintf(f2,"%s\n","     echo \" </head><body><span style='font-size:20pt; font-weight:700;'><font color=\\\"green\\\">\\n\";");
                sprintf(str,"echo \" %s\";",Memo1->Lines->Strings[1]);
                fprintf(f2,"%s\n",str);
                fprintf(f2,"%s\n"," echo \" </font>\\n\";");
        if (que>0)
                fprintf(f2,"%s\n","   echo \"\\n <br/> Время ответа \"; echo (int)(($t=time()-$TIME)/60); echo \" мин \";  echo ($t%60).\" сек\";");
        else

                fprintf(f2,"%s\n"," echo \" \\n <br/> Время ответа \"; echo (int)(($t=time()-$TIME)/60); echo \" мин \";  echo ($t%60).\" сек\";");
        //          fprintf(f2,"%s\n","$R="__HOST__"; unlink(\"/tmp/\".$R);");

                fprintf(f2,"%s\n", " echo \" \\n <br/> Задано вопросов \"; echo $QS;");
                fprintf(f2,"%s\n", " echo \" \\n <br/> Правильных ответов \"; echo $NB; echo \"<br/>\";");
                switch(mark_form){
                case 0:
                fprintf(f2,"if($NB>=%d)echo \"<span style={color:red}> ЗАЧТЕНО</span>\"; else echo \" НЕ ЗАЧТЕНО\"" ,mark_ok);
                break;
                case 1:
                fprintf(f2,"if($NB>=%d)echo \"<span style={color:red}> ОТЛИЧНО</span>\"; else " ,mark_5);
                fprintf(f2,"if($NB>=%d)echo \"<span style={color:blue}> ХОРОШО</span>\"; else " ,mark_4);
                fprintf(f2,"if($NB>=%d)echo \"<span style={color:green}> УДОВЛЕТВОРИТЕЛЬНО</span>\"; else " ,mark_3);
                fprintf(f2,"echo \"<span style={color:black}> ПЛОХО</span>\"; " );
                break;
                }
                fprintf(f2, " ?> <br/><br/><br/> <center><a href=\"javascript:close();\">ВЫХОД</a></center></span> </body></html>");
                fclose(f2);
                break;
         }// for(;;)
        } } }
```