МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования Московский государственный университет технологий и управления

имени К.Г. Разумовского.

Университетский колледж информационных технологий.

Специальность: 230115 «Программирование в компьютерных системах»

КУРСОВОЙ ПРОЕКТ

По дисциплине: «Технология разработки и защиты баз данных.»

|  |  |
| --- | --- |
| На тему: «*Программа контроля знаний учащихся*» |  |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  УниКИТ.230115.010ПЗ |  |

Г

Группа П-303

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.Ю. Захаров

Студент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.А. Ларионова

Руководитель

Москва 2014

**Содержание:**

***Оглавление***

[Введение. 4](#_Toc388856492)

[1. Общая часть. 9](#_Toc388856493)

[*1.1. Обзор и анализ предметной области. 9*](#_Toc388856494)

[*1.2. Жизненный цикл базы данных. 11*](#_Toc388856495)

[*1.3. Защита баз данных. 13*](#_Toc388856496)

[*1.4. Выбор и характеристика СУБД. 16*](#_Toc388856497)

[*1.5. Выбор и характеристика среды разработки приложения. 17*](#_Toc388856498)

[2. Специальная часть. 18](#_Toc388856499)

[*2.1. Постановка задачи. 18*](#_Toc388856500)

[*2.2. Логическая схема базы данных. 19*](#_Toc388856501)

[*2.3. Описание таблиц. 20*](#_Toc388856502)

[*2.4. Другие объекты базы данных (запросы, отчёты, хранимые процедуры, триггеры) 24*](#_Toc388856503)

[*2.5. Разработка приложения 31*](#_Toc388856504)

[*2.5.1. Структурная схема приложения. 31*](#_Toc388856505)

[*2.5.2. Разработка интерфейса пользователя. 31*](#_Toc388856506)

[*2.5.3. Описание процесса отладки приложения. 32*](#_Toc388856507)

[*2.6. Выбор способа защиты базы данных. 32*](#_Toc388856508)

[*2.7. Инструкция пользователю. 33*](#_Toc388856509)

[*2.7.1. Общие сведения об информационной системе. 33*](#_Toc388856510)

[*2.7.2. Требования к техническим средства. 33*](#_Toc388856511)

[*2.7.3. Требования к программным средствам. 33*](#_Toc388856512)

[*2.7.4. Настройка информационной системы. 33*](#_Toc388856513)

[*2.7.5. Формы ввода. 33*](#_Toc388856514)

[*2.7.6. Отчеты. 37*](#_Toc388856515)

[*2.7.7. Защита информационной системы. 38*](#_Toc388856516)

[3. Спецификация. 38](#_Toc388856517)

[Заключение. 41](#_Toc388856518)

[Приложение. (Код программы) 42](#_Toc388856519)

[Используемая литература: 82](#_Toc388856520)

# **Введение.**

Курсовой проект посвящен: приобретению навыков по разработке удалённой базы данных и приложения для работы с ней с использованием объектно-ориентированного и визуального программирования, а также, непосредственно, разработке программы контроля знаний учащихся.

Тема курсового проекта, является актуальной, так как грамотно составленная программа позволяет тестировать студентов и составлять различного рода отчёты по проведённым тестированиям, что позволяет сэкономить много времени, бумажных ресурсов, а также дает мгновенную и объективную оценку результатов по тестированию.

В настоящее время в учебный процесс активно внедряются компьютерные технологии, в следствии чего получили большое распространение программы для электронного тестирования учащихся. Такие технологии являются наиболее удобными для преподавателя, поскольку, данный подход позволяет сэкономить кучу времени, а главное, критерии оценивания являются самыми объективными, так как машина не способна умышленно завышать, или занижать баллы. Ещё одной особенностью такого программного обеспечения является то, что оно изначально может быть ориентировано под конкретную дисциплину, или быть универсальным (то есть, тестировать можно проходить по любой дисциплине).

Обзор аналогов:

1. Редактор для создания электронных тестов EasyQuizzy (коммерческий ПП).

Рассмотрим пример программы для тестирования на основе платной программы EasyQuizzy.

EasyQuizzy - это программа, с помощью которой можно создавать и редактировать компьютерные тесты. Используя подготовленные тесты, можно облегчить работу преподавателей при проведении ежедневных "быстрых" опросов учащихся и ускорить проверку результатов, т.к. выставление оценки происходит автоматически на основе системы оценивания, выбранной при создании теста.

Программа EasyQuizzy предоставляет единый последовательный интерфейс для создания тестов. Начинается создание теста со вкладки "Информация о тесте", где требуется сформулировать название и описание теста, а также указать автора теста. Затем нужно перейдите на вкладку "Вопросы и ответы". Добавление вопросов, вариантов ответов и последующий выбор правильных вариантов ответов не вызывает затруднений. В текст вопросов и вариантов ответов можно вставлять картинки, формулы и специальные символы.

После формирования списка вопросов, на вкладке "Настройки теста" идет настройка параметров теста: количество задаваемых вопросов и их порядок, система оценивания, ограничение времени и формат итогового отчета.

При сохранении теста формируется независимая программа, которую достаточно скопировать на любой компьютер и запустить, чтобы начать тестирование.

Интерфейс тестов спроектирован так, чтобы при прохождении теста пользователь задумывался только над задаваемыми вопросами, а не над тем, как работать с программой.

Для ответа на вопрос достаточно выбрать правильный ответ (ответы) и подтвердить выбор.

По окончании тестирования выставляется рекомендуемая оценка.

Также имеется возможность распечатать итоговый отчет или сохранить его и отправить по электронной почте.

При составлении теста можно использовать следующие типы вопросов:

1. Альтернативный выбор.
2. Выбор одного правильного ответа.
3. Выбор нескольких правильных ответов.
4. Установление правильной последовательности.
5. Установление соответствия
6. Свободный ответ.

В каждом тесте можно свободно совмещать вопросы всех трех типов.

Редактор тестов позволяет вставлять рисунки в любом формате и менять их размеры. При наличии на компьютере установленного пакета Microsoft Office 95 или более поздней версии, возможна вставка в тест формул Microsoft Equation.

На компьютерах тестируемых пользователей для отображения формул при запуске теста наличие пакета Microsoft Office не требуется.

Созданные тесты совместимы с операционными системами:

· Windows 2000;

· Windows XP;

· Windows Fundamentals for Legacy PC;

· Windows 2003 Server;

· Windows Vista;

· Windows Embedded;

· Windows 2008 Server;

· Windows 7;

а также:

· Linux, FreeBSD, Mac OS X и Solaris с установленным Wine

Также требуется:

· не менее 3 Мбайт свободного места на диске;

· для возможности редактирования формул с помощью Microsoft Equation необходим установленный пакет Microsoft Office 95 или более новая версия.

У данной программы удобный интерфейс и довольно много разных функций. Но помимо преимуществ, данная программа имеет свои **недостатки**:

1. Это высокая цена по сравнению с аналогами (400 рублей).

2. Не смотря на большой перечень операционных систем, последние версии данной программы работают далеко не на каждой из них.

1. Автоматизированная информационная система контроля знаний Easy Test. (коммерческий ПП)

Автоматизированная информационная система контроля знаний (АИСКЗ в дальнейшем) EasyTest используются во многих областях, например:

· при проведении пробных ЕГЭ;

· при принятии теоретического экзамена в ГАИ;

· в различных тестах в Интернете (учебные, социологические опросы и т.д.).

АИСКЗ имеют ряд преимуществ по сравнению с традиционными методами контроля:

1. Процесс проверки заданий автоматизирован.

2. Выставление оценок происходит на основании количества правильно выполненных заданий теста.

3. Контроль знаний можно проводить на каждом занятии, так как такой контроль при малом количестве вопросов, выполняется очень быстро.

4. Оценка за знания ставится объективно и не зависит от преподавателя.

5. Все результаты проведенных тестов сохраняются, и всегда можно повторно вернуться к результатам.

6. У преподавателя всегда есть статистика по успеваемости учащихся.

7. Высокий уровень масштабируемости тестовых систем.

8. Хорошая защита от фальсификации результатов тестирования.

9. Возможность дистанционной проверки знаний учащегося находящегося, вне учебного заведения (например, по причине болезни)

Чтобы обеспечить совместимость с различными типами компьютеров и различными ОС, в качестве рабочей среды АИСКЗ был выбран Интернет (тип приложения - Web-приложение). Для работы Web-приложения нужен Web-браузер, который по умолчанию всегда устанавливается в современных операционных системах, поэтому преимущество такой схемы очевидно. Отсутствие Интернета не мешает использовать АИСКЗ, так как в любом учебном заведении есть локальная вычислительная сеть, и АИСКЗ может быть установлена на Web-сервер этой сети.

Для новой АИСКЗ были определены следующие требования:

1. Информационная система должна функционировать практически на любом компьютере и с любой ОС.

2. Информационная система должна иметь простой, понятный и удобный интерфейс.

3. Информационная система должна работать стабильно, гарантировать сохранность результатов тестирования.

4. Информационная система должна быть легко настраиваема, должна иметь установщик системы.

5. Информационная система должна использовать общую базу для хранения всех настроек системы.

6. Информационная система должна обеспечивать возможность масштабирования и инвариантности тестов.

7. В информационной системе должны быть реализованы механизмы визуализации результатов тестирования.

8. Информационная система должна быть защищена от возможности несанкционированного доступа.

9. Информационная система должна содержать простой механизм регистрации новых участников.

10. Информационная система должна иметь возможность демонстрации ошибок для участников тестов.

11. В информационной системе должна быть возможность размещения дополнительных учебных материалов.

12. В информационной системе должен быть электронный журнал с регистрацией имени, дня и времени посещения. Также в журнале должно быть записано время прохождения теста.

Для начала работы с АИСКЗ предварительно необходимо пройти регистрацию, перейдя по ссылке "Регистрация”. Необходимо ввести логин, пароль (2 раза) и адрес электронной почты, затем нажать кнопку "Готово”. Через некоторое время на указанный адрес электронный почты должно прийти сообщение со ссылкой на регистрацию нового пользователя. После перехода по ссылке из письма новый пользователь будет активирован.

После выполнения регистрации можно зайти в систему АИСКЗ, используя данные регистрации.

После входа появляется главная страница АИСКЗ.

Слева находится главное меню тестирующей системы. В нем содержится 3 пункта:

1. Просмотр тестов - здесь можно выбрать нужный тест и приступить к его выполнению.

2. Создание теста - этот пункт меню предназначен для создания нового теста.

3. Оценки - здесь можно узнать оценки за пройденные тесты.

Чтобы приступить к выполнению теста необходимо перейти на страницу "Просмотр тестов” и напротив нужного теста щелкнуть по ссылке "Начать”. После прохождения теста результат можно посмотреть на странице "Оценки”.

Данная программа имеет лишь один **недостаток**: очень высокая цена (1000 рублей на 1 ПК).

1. Программа для созданий тестов и тестовой проверки уровня знаний x-TLS. (Бесплатная)

Программа x-TLS представляет собой современную инструментальную среду для создания автоматизированных обучающих и контролирующих систем на основе расширенных мультимедийных тестовых заданий.

Система полностью бесплатна и распространяется по лицензии "x-TLS лицензионные условия".

Техническая база:

x-TLS представляет собой полностью клиент-серверную кроссплатформенную среду, теоретически способную работать в кластере.

Серверная часть написана с использованием реляционной СУБД mySQL и технологий java-servlets. В качестве клиентского ПО студенческого рабочего места используется любой современный браузер (Mozilla FireFox, Miscrosoft Internet Explorer, Opera и т.д.).

Рабочее место разработчика мультимедийных тестов написано с использованием языка C++ и представляет собой WYSIWYG-среду.

Данные особенности позволяют значительно повысить производительность системы и снизить системные требования, а также затраты на развертывание системы в условиях учебного заведения, по сравнению с использованием php или технологий "толстого клиента".

Системные требования:

Серверная часть: любая операционная система с поддержкой Java SE и существующей реализацией MySQL 5 (MS Windows, Linux, FreeBSD, Solaris и т.д.), 128мб ОЗУ и выше, 1 гб места на диске.

Модуль редактора: MS Windows 2000 и выше.

Модуль студента и модуль экспресс-администрирования: любая платформа, позволяющая развернуть современный браузер (Internet Explorer 6 и выше, Mozilla FireFox 1.0 и выше и другие.)

**Плюсы:**

1. Бесплатная.
2. Клиент серверная программа.

**Минусы:**

1. Мало функций.

В ходе проектирования и разработки программного продукта, должны быть реализованы следующие функции:

1. Регистрация учащихся перед выполнением теста. (Ввод имени, фамилии, отчества, группы, номера студенческого билета.)
2. Вход учащегося в его личное дело. (Ввод фамилии, номера студенческого билета.)
3. Выполнение учащимся тестирования.
   1. Выбор дисциплины из списка дисциплин.
   2. Выбор темы из списка тем по заданной дисциплине.
   3. Чередование вопросов в произвольном порядке (дублирование вопросов исключается).
   4. Возможность выбора ответа из предложенного списка (не менее 3-х) с подсчетом количества набранных очков.
   5. Возможность сохранения результатов проведенного теста (фамилия, группа, дисциплина, тема, дата, время начала, время конца, сумма баллов, оценка).
   6. Сохранение выбранного учащимся ответа и правильного ответа на каждый вопрос в таблице для решения «спорных моментов».
   7. Выдача резюме и оценки в конце теста.
   8. Выдача отчета в конце теста. (Фамилия, группа, номер вопроса, номер выбранного учащимся ответа, номер правильного ответа, дисциплина, тема.)
4. Регистрация преподавателей: фамилия, имя, отчество, логин, пароль. (Для подтверждения личности учителя администратору требуется ввести свой логин и пароль.)
5. Вход преподавателей в информационную систему и разрешение на:
   1. Просмотр результатов проведенного теста (с учетом одного или нескольких критериев одновременно):
      1. По дате.
      2. По конкретному учащемуся.
      3. По конкретной группе.
      4. По конкретной теме, конкретной дисциплины.
   2. Получение ведомости/отчета по результатам тестирования конкретной группы по конкретной дисциплине.
   3. Получение отчета по итогам теста (с перечнем вопросов и ответов) для решения спорных вопросов.
   4. Очистку журнала регистрации студентов (всех или выбранных).
   5. Ввод новых дисциплин, тем, вопросов, и ответов по ним.
   6. Установление и изменение критериев оценки с автоматическим пересчетом и заменой баллов за вопросы.

Выполнение курсового проекта предполагает разбиение работы на этапы с отведенными для них сроками выполнения.

**Основные этапы разработки проекта:**

* Разработка концептуальной схемы базы данных.
* Разработка логической схемы базы данных и преобразование её в физическую модель MS SQL Server средствами ERWIN Modeler.
* Разработка приложения.

# **Общая часть.**

# **Обзор и анализ предметной области.**

**Тест** (от английского слова test - проверка, задание) - это система заданий, позволяющая измерить уровень усвоения знаний, степень развития определенных психологических качеств, способностей, особенностей личности (в зависимости от тематики теста). Введение тестового контроля существенно повышает мотивацию обучения и заинтересованность обучаемого. Тестирование на компьютере более интересно по сравнению с традиционными формами опроса, что создает положительную мотивацию у студентов. Тестовый блок заданий позволяет получить весьма информативный результат о знаниях учащегося, благодаря большему количеству вопросов и задач, охватывающий изучаемый материал, чем обычные стандартные методы проверки знаний в виде контрольных работ, при той же затрате времени. Задания, предложенные в тестовом блоке стимулируют учащегося на развитие и применение своих интеллектуальных способностей. Тестовый контроль знаний можно проводить как традиционно "бумажным" способом так и с применением компьютера в локальной сети.

Тесты можно разделять по многим признакам. Основная классификация — по направленности теста: оценивать можно способности человека, его личностные качества, интеллект, отдельные психические функции (внимание, память, воображение), знания. Поскольку курсовой проект посвящен вопросам использования тестов в образовании, то дальше речь пойдет именно о последних - тестах знаний.

Виды тестовых заданий:

Рассмотрим наиболее популярную классификацию тестовых заданий. В рамках данной классификации тестовые задания можно разделить на две группы:

1. Тестовые задания закрытого типа (каждый вопрос сопровождается готовыми вариантами ответов, из которых необходимо выбрать один или несколько правильных);
2. Тестовые задания открытого типа (на каждый вопрос испытуемый должен предложить свой ответ: дописать слово, словосочетание, предложение, знак, формулу и т.д.).

Выбор типа и вида тестового задания определяется, прежде всего, целями, в соответствии с которыми проводится тестирование, характером материала, усвоение которого необходимо выявить, возрастными особенностями испытуемых. Немаловажное значение играет финансовое, кадровое и ресурсное обеспечение, а также запас времени, которым располагает разработчик.

Виды вопросов:

Множественный выбор - испытуемому необходимо выбрать один или несколько правильных ответов из приведенного списка.

Альтернативный выбор - испытуемый должен ответить «да» или «нет».

Установление соответствия - испытуемому предлагается установить соответствие элементов двух списков.

Установление последовательности - испытуемый должен расположить элементы списка в определенной последовательности.

Свободное изложение - испытуемый должен самостоятельно сформулировать ответ; никакие ограничения на них в задании не накладываются.

Дополнение - испытуемый должен сформулировать ответы с учетом предусмотренных в задании ограничений (например, дополнить предложение).

В последние годы в сфере образования наблюдается увеличение интереса к автоматизации всех процессов обучения, в том числе и к процессу контроля знаний учащихся. Программы автоматического компьютерного тестирования вызывают высокий интерес из-за удобства и скорости прохождения тестирования, а также мгновенной оценке результатов. Самым популярным видом такого контроля является тестирование, основанное на диалоге вычислительной системы с пользователем. Компьютерное тестирование позволяет ускорить проведение контроля знаний и существенно облегчить обработку результатов, а также существенно снизить нагрузку на преподавателя, увеличить заинтересованность и повысить доверительную обстановку в аудитории со стороны учащихся.

Целью данной курсовой работы является разработка, тестирование и поддержка программного комплекса: " Программа контроля знаний учащихся."

Основные принципы предметной области:

1. Регистрация учащегося перед выполнением теста.

После входа в информационную систему учащемуся предлагается выбор: зайти в свое личное дело (ввести фамилию и номер студенческого билета) или же зарегистрироваться в системе. Регистрация проводится путем ввода своих личных данных в поля и после окончания регистрация вся информация вносится в базу данных, уже после чего учащийся может зайти в свое личное дело и продолжить работу.

1. Вход учащегося в своё личное дело, выбор и прохождение тестирования.

После входа, учащегося в его личное дело он может ознакомиться со своими личными данными, выбрать дисциплину и тему по заданной дисциплине, и пройти тестирование. Все выбранные варианты ответов учащегося вносятся в базу данных по мере прохождения теста, а также после окончания тестирования ему выдается резюме, оценка и количество полученных баллов. Также учащийся может ознакомиться с отчетом, где указаны его ошибки, и распечатать его.

1. Регистрация преподавателя перед выполнением теста.

После входа в информационную систему преподавателю предлагается выбор: зайти в свою учетную запись (ввести логин и пароль) или же зарегистрироваться в системе. Регистрация проводится путем ввода своих личных данных в поля и после окончания регистрация вся информация вносится в базу данных при учете, что данное действие было подтверждено администратором базы данных. После этого преподаватель может зайти в свою учетную запись и продолжить работу.

1. Вход преподавателя в учетную запись и работа под ней.

После входа в учетную запись преподаватель может выполнять следующие функции: просмотреть результаты проведенных тестов по одному или нескольким критериям (по дате; учащемуся; группе; конкретной теме, конкретной дисциплины), получить ведомость/отчет по результатам тестирования конкретной группы по конкретной дисциплины, получить отчет по итогам теста (с перечнем вопросов и ответов) для решения спорных вопросов, очистить журнал регистрации учащихся, установить и изменить критерии оценки.

1. Административные функции.

Администратор системы обладает дополнительными функциями: регистрация учителей в системе, редактирование таблиц: учащиеся, учителя.

Для обработки результатов будет использована клиент-серверная архитектура, также будет реализована работа одновременно нескольких пользователей. Данная схема будет позволять регистрировать клиентов одновременно из call-центра и на стойке регистрации. Для развития данной системы следует использовать

трёхзвенную архитектуру с разработкой Internet приложения.

Основными объектами в тестировании являются: учащийся (тот кто проходит тестирование), преподаватель (тот кто просматривает результаты, вводит новые дисциплины, темы, вопросы, ответы, вводит и редактирует критерии оценки), дисциплины (один из параметров выбора вопросов к тесту), темы по дисциплинам (второй из параметров выбора вопросов к тесту), вопросы и ответы (сами тесты: то, на что отвечает учащийся и что оценивается в результате ответа), оценки (критерии оценивания результатов), результаты (оценка и анализ правильных и не правильных ответов определенного учащегося).

# **Жизненный цикл базы данных.**

1. Стадия анализа – на этой стадии будет производится анализ предметной области и выявляться требования к ней. Так же будет даваться оценка актуальности разработки. (п. 1.1.)
2. Стадия проектирования – на этой стадии будет создана логическая структура базы данных, функциональное описание программных модулей и информационных запросов. (п. 2.1. – 2.4.)
3. Стадия реализации – на этой стадии решаются задачи по разработке программного доступа к базе данных, проводится тестирование. (п. 2.5. – 2.6.)
4. Стадия эксплуатации и сопровождения. (п. 2.7.)

Жизненный цикл базы данных состоит из следующих этапов:

**1. Предварительное планирование** – планирование БД, выполняемое в процессе разработки стратегического плана БД. В процессе планирования собирается следующая информация:

• Какие прикладные программы используются, и какие функции они выполняют;

• Какие файлы связаны с каждым из этих приложений;

• Какие новые приложения и файлы находятся в процессе работы.

Данная информация помогает определить, как используется информация приложений, определить будущие требования к системе БД.

Информация этого этапа документируется в виде обобщенной модели данных.

**2. Проверка осуществимости**- здесь определяется технологическая, операционная и экономическая осуществимость плана создания БД, т. е.:

• Технологическая осуществимость – есть ли технология для реализации запланированной БД?

• Операционная осуществимость – есть ли средства и эксперты, необходимые для успешного осуществления плана создания БД?

• Экономическая целесообразность – можно ли определить выводы? Окупится ли запланированная система? Можно ли оценить издержки и выгоду?

**3. Определение требований** включает выбор целей БД, выяснение информационных требований к системе и требований к оборудованию и программному обеспечению. Таким образом, на данном этапе сбора данных и определения требований создаётся общая информационная модель, выражающаяся в следующих задачах:

• Определяются цели системы путём анализа информационных потребностей. Здесь также обязательно указывается, какую именно БД следует создавать (распределённую, целостную) и какие коммуникационные средства необходимы. Выходной документ – комментарий, описывающий цели системы.

• Определение пользовательских требований: документация в виде обобщённой информации (комментарии, отчёты, опросы, анкеты и т. д.); фиксация функций системы и определение прикладных систем, которые будут выполнять эти требования. Данные представляются в виде соответствующих документов.

• Определение общих требований к оборудованию и программному обеспечению, связанных с поддержанием желаемого уровня быстродействия. (Выяснение количества пользователей системы, числа входных сообщений в день, количество распечаток). Данная информация используется для выбора типов компьютеров и СУБД, объёма дисков, количества принтеров. Данные этого этапа излагаются в отчёте, содержащем примерные конфигурации оборудования и программного обеспечения.

• Разработка плана поэтапного создания системы, включающий выбор исходных приложений.

**4. Концептуальное проектирование** – создание концептуальной схемы БД. Спецификации разрабатываются в той степени, которая необходима для перехода к реализации.

Основным выходным документом является единая инфологическая модель (или схема БД на концептуальном уровне). При разработке данной модели используются информация и функции, которые должна выполнить система, определённые на этапе сбора и определения требований к системе. На данном этапе желательно также определить: 1) правила для данных; 2) правила для процессов; 3) правила для интерфейса.

**5. Реализация** – процесс превращения концептуальной модели в функциональную БД. Он включает в себя следующие этапы.

1) Выбор и приобретение необходимой СУБД.

2) Преобразование концептуальной (инфологической) модели БД в логическую и физическую модель данных:

• На основе инфологической модели данных строится схема данных для конкретной СУБД, при необходимости реализуется денормализация БД с целью ускорения обработки запросов во всех критичных по времени приложениях;

• Определяются, какие прикладные процессы необходимо реализовать в схеме данных как хранимые процедуры;

• Реализовать ограничения, предназначенные для обеспечения целостности данных и реализации правил для данных;

• Спроектировать и сгенерировать триггеры для реализации всех централизованно определённых правил для данных и правил целостности данных, которые не могут быть заданы как ограничения;

• Разработать стратегию индексирования и кластеризации; выполнить оценку размеров всех таблиц, кластеров и индексов;

• Определить уровни доступа пользователей, разработать и внедрить правила обеспечения безопасности и аудита. Создать роли и синонимы для обеспечения многопользовательского доступа с согласованными уровнями полномочий доступа.

• Разработать сетевую топологию БД и механизм бесшовного доступа к удалённым данным (реплицированная или распределённая БД).

3) Построение словаря данных, который определяет хранение определений структуры данных БД. Словарь данных также содержит информацию о полномочиях доступа, правилах защиты данных и контроля данных.

4) Заполнение базы данных.

5) Создание прикладных программ, контроль управления.

6) Обучение пользователей.

**6. Оценка и усовершенствование схемы БД.** Включает опрос пользователей с целью выяснения функциональных неучтенных потребностей. При необходимости вносятся изменения, добавление новых программ и элементов данных по мере изменения и расширения потребностей.

**Таким образом, ЖЦБД включает в себя:**

• Изучение предметной области и представление соответствующей документации (1-3).

• Построение инфологической модели (4).

• Реализация (5).

• Оценка работы и поддержка БД (6).

# **Защита баз данных.**

Защита данных включает предупреждение случайного или несанкционированного доступа к данным, их изменения или разрушения со стороны пользователей или при сбоях аппаратуры. Реализация защиты включает:

* контроль достоверности данных с помощью ограничений целостности;
* обеспечение безопасности данных (физической целостности данных);
* обеспечение секретности данных.

Методы защиты баз данных в различных СУБД несколько отличаются друг от друга. Анализ современных СУБД показывает, что методы защиты условно делятся на две группы: основные и дополнительные.

К основным средствам защиты относится:

* Защита паролем;

Защита паролем представляет собой простой и эффективный способ защиты БД от несанкционированного доступа. Пароли устанавливаются пользователями или администраторами БД. Учет и хранение паролей выполняется самой СУБД. Обычно, пароли хранятся в определенных системных файлах СУБД в зашифрованном виде. После ввода пароля пользователю СУБД предоставляются все возможности по работе с БД. Парольная защита является достаточно слабым средством, особенно если пароль не шифруется.

* Шифрование данных и программ;

Шифрование – это преобразование читаемого текста в нечитаемый текст, при помощи некоторого алгоритма; применяется для защиты уязвимых данных.

Процесс дешифрования восстанавливает данные в исходное состояние.

* Разграничение прав доступа к объектам базы данных;

В целях контроля использования основных ресурсов СУБД во многих системах имеются средства установления прав доступа к объектам БД. Права доступа определяют возможные действия над объектами. Владелец объекта (пользователь, создавший объект), а также администратор БД имеют все права. Остальные пользователи к разным объектам могут иметь различные уровни доступа. Разрешение на доступ к конкретным объектам базы данных сохраняется в файле рабочей группы.

Файл рабочей группы содержит данные о пользователях группы и считывается во время запуска. Файл содержит следующую информацию: имена учетных записей пользователей, пароли пользователей, имена групп, в которые входят пользователи.

По отношению к таблицам могут предусматриваться следующие права доступа:

1. просмотр (чтение) данных;
2. изменение (редактирование) данных;
3. добавление новых записей;
4. добавление и удаление данных;
5. изменение структуры таблицы.

* защита полей и записей таблиц БД.

Применительно к защите данных в полях таблиц можно выделить следующие уровни прав доступа:

1. полный запрет доступа;
2. только чтение;
3. разрешение всех операций (просмотр, ввод новых значений, удаление и изменение).

К дополнительным средствам защиты БД можно отнести такие, которые нельзя прямо отнести к средствам защиты, но которые непосредственно влияют на безопасность данных.

К дополнительным средствам защиты относятся:

• встроенные средства контроля значений данных в соответствии с типами;

Редактируя БД, пользователь может случайно ввести такие значения, которые не соответствуют типу поля, в которое это значение вводится (например, ввод в числовое поле текстовой информации). В этом случае СУБД с помощью средств контроля значений блокирует ввод и сообщает пользователю об ошибке.

• повышения достоверности вводимых данных;

Средства повышения достоверности вводимых значений в СУБД служат для более глубокого контроля, связанного с семантикой обрабатываемых данных. Они обычно обеспечивают возможность при создании таблицы указывать следующие ограничения на значения: минимальное и максимальное значения, значение, принимаемое по умолчанию (если нет ввода), требование обязательного ввода; задание маски (шаблона) ввода и т.д.

Более совершенной формой организации контроля достоверности информации в БД является разработка хранимых процедур. Механизм хранимых процедур применяется в БД, размещенных на сервере. Сами хранимые процедуры представляют собой программы, алгоритмы которых предусматривают выполнение некоторых функций (в том числе контрольных) над данными. Процедуры хранятся вместе с данными и при необходимости вызываются из приложений либо при наступлении некоторых событий в БД.

• обеспечения целостности связей таблиц;

Решение прикладной задачи, как правило, требует выбора информации из нескольких таблиц. Таблицы в базе данных могут быть связаны. Функции поддержания логической целостности связанных таблиц берет на себя СУБД. Если СУБД не реализует эти функции, то ответственность за корректность связей возлагается на приложение.

• организации совместного использования объектов БД в сети.

В распределенных информационных системах, работающих с базами данных, возникает проблема разрешения конфликтов между различными действиями над одними и теми же объектами (совместного использования объектов БД). Например, что делать в случае, когда один из пользователей локальной сети редактирует БД, а другой хочет изменить ее структуру? Для таких ситуаций в СУБД должны быть предусмотрены механизмы разрешения конфликтов.

Обычно при одновременной работе нескольких пользователей в сети используются блокировки. Блокировки могут действовать на различные объекты БД и на отдельные элементы объектов. Блокировки объектов возникают, когда параллельно с использованием объекта предпринимается попытка входа в режим разработки этого же объекта. Применительно к таблицам баз данных дополнительные блокировки могут возникать при работе с отдельными записями или полями.

# **Выбор и характеристика СУБД.**

В качестве СУБД для выполнения курсового проекта выбрана среда MS SQL Server 2008 r2.

Microsoft SQL Server — мощная система управления реляционными базами данных (СУРБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями, реализует объёмный набор встроенных функций, функций безопасности, имеет большое количество объектов, направленных на контроль данных, оптимизацию работы с данными, реализует удалённый доступ к базе, одновременную работу с базой с нескольких рабочих позиций, среда имеет простую конфигурацию и настройку, что упрощает её распространение и работу с ней. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.

SQL Server 2008 R2 обеспечивает интеграцию с Microsoft Office, гарантирует повышенную безопасность и производительность средств разработки, масштабируемость и высокая доступность.

**Важнейшие характеристики данной СУБД - это:**

1. Простота администрирования.
2. Возможность подключения к Web.
3. Быстродействие и функциональные возможности механизма сервера СУБД,

наличие средств удаленного доступа.

В комплект средств административного управления данной СУБД входит целый набор специальных мастеров и средств автоматической настройки параметров конфигурации. Также данная БД оснащена замечательными средствами тиражирования, позволяющими синхронизировать данные ПК с информацией БД и наоборот. Входящий в комплект поставки сервер OLAP дает возможность сохранять и анализировать все имеющиеся у пользователя данные. В принципе данная СУБД представляет собой современную полнофункциональную база данных, которая идеально подходит для малых и средних организаций. Необходимо заметить, что SQL Server уступает другим рассматриваемым СУБД по двум важным показателям: программируемость и средства работы. При разработке клиентских БД приложений на основе языков Java, HTML часто возникает проблема недостаточности программных средств SQL Server и пользоваться этой СУБД будет труднее, чем системами DB2, Informix, Oracle или Sybase. Общемировой тенденцией в XXI веке стал практически повсеместный переход на платформу LINUX, а SQL Server функционирует только в среде Windows. Поэтому использование SQL Server целесообразно, только если для доступа к содержимому БД используется исключительно стандарт ODBC, в противном случае лучше использовать другие СУБД.

# **Выбор и характеристика среды разработки приложения.**

В качестве среды разработки будет использована Embarcadero RAD Studio XE5.

Embarcadero RAD Studio — среда быстрой разработки приложений (RAD) для Microsoft Windows фирмы Embarcadero Technologies.

Текущая версия Embarcadero RAD Studio XE5 объединяет Delphi XE5 и C++ Builder XE5 в единую интегрированную среду разработки.

Данная среда и язык Embarcadero C++ Builder выбрана из-за:

1. Уникальной совокупности простоты языка и генерации машинного кода, которая позволяет непосредственно, и, при желании, достаточно низкоуровнево взаимодействовать с операционной системой, а также с библиотеками, написанными на C/C++ и других языках программирования;
2. Создания программ, которые не зависимы от стороннего ПО, например, от Microsoft .NET Framework или Java Virtual Machine;
3. Выделение и освобождение памяти, которое контролируется в основном пользовательским кодом, что, с одной стороны, ужесточает требования к качеству кода, а с другой — делает возможным создание сложных приложений, с высокими требованиями к отзывчивости (работа в реальном времени).

Таким образом данная среда позволяет просто, эффективно и качественно разрабатывать различные эргономичные, многофункциональные, оптимальные приложения.

C++ Builder— программный продукт, инструмент быстрой разработки приложений (RAD), интегрированная среда программирования (IDE), система, используемая программистами для разработки программного обеспечения на языке программирования C++.

C++ Builder объединяет в себе комплекс объектных библиотек (STL, VCL, CLX, MFC и др.), компилятор, отладчик, редактор кода и многие другие компоненты. Цикл разработки аналогичен Delphi. Большинство компонентов, разработанных в Delphi, можно использовать и в C++ Builder без модификации, но обратное утверждение не верно.

Borland C++Builder,сегодня является наиболее совершенной визуальной средой быстрой разработки на С++ для Windows. В ее состав входит около 200 самых разных компонентов, а создание законченной программы требует минимума усилий. Ближайший конкурент Borland C++Builder — это не система Microsoft Visual C++, которая построена по другой схеме, a Microsoft Visual Basic. Однако эффективность программ, создаваемых с помощью C++Builder, в десятки раз превосходит быстродействие программ, написанных на MS Visual Basic. Да и по числу свободных доступных компонентов равных среде C++Builder сегодня не найти.

C++Builder поддерживает основные принципы объектно-ориентированного программирования — инкапсуляцию, полиморфизм и множественное наследование, а также нововведенные спецификации и ключевые слова в стандарте языка C++.

# **Специальная часть.**

# **Постановка задачи.**

Средствами MS SQL SERVER спроектировать базу данных и создать приложение средствами Embarcadero RAD Studio (C++ Builder), позволяющее, контролировать знания учащихся посредством тестирования. Каждый учащийся определенного учебного заведения может зарегистрироваться в системе и войти в свое личное дело, где он может ознакомиться с личной информацией и пройти тестирование, выбрав перед этим дисциплину из списка дисциплина и тему по заданной дисциплине. После прохождения теста учащийся может ознакомиться с своим резюме, набранными баллами и оценкой, а также ознакомиться с своими ошибками посредством отчета и распечатать его. Преподаватели учебного заведения могут, после входа в систему под своей учетной записью, ознакомиться с результатами тестирования, получить отчеты по тестированию конкретной группы конкретной дисциплины и отчеты по итогам тестирования, очистить журнал регистрации студентов, ввести новые темы, дисциплины, вопросы и ответы, установить или изменить критерии оценки. Необходимо обеспечить ввод, редактирование и просмотр данных в удобной для пользователя форме.

Система должна обеспечивать проведения тестирования студентов, выполнение всех перечисленных функций для преподавателей и предоставлять возможность просматривать следующую информацию и генерировать следующие документы:

1. Отчет, после прохождения студентом тестирования, о его ошибках, допущенных в тесте.
2. Отчет по результатам тестирования конкретной группы по конкретной дисциплине.
3. Отчет по итогам теста (для решения спорных вопросов).

В ходе проектирования и разработки программного продукта, должны быть реализованы следующие функции:

1. Регистрация учащихся перед выполнением теста. (Ввод имени, фамилии, отчества, группы, номера студенческого билета.)

2. Вход учащегося в его личное дело. (Ввод фамилии, номера студенческого билета.)

3. Выполнение учащимся тестирования.

3.1. Выбор дисциплины из списка дисциплин.

3.2. Выбор темы из списка тем по заданной дисциплине.

3.3. Чередование вопросов в произвольном порядке (дублирование вопросов исключается).

3.4. Возможность выбора ответа из предложенного списка (не менее 3-х) с подсчетом количества набранных очков.

3.5. Возможность сохранения результатов проведенного теста (фамилия, группа, дисциплина, тема, дата, время начала, время конца, сумма баллов, оценка).

3.6. Сохранение выбранного учащимся ответа и правильного ответа на каждый вопрос в таблице для решения «спорных моментов».

3.7. Выдача резюме и оценки в конце теста.

3.8. Выдача отчета в конце теста. (Фамилия, группа, номер вопроса, номер выбранного учащимся ответа, номер правильного ответа, дисциплина, тема.)

4. Регистрация преподавателей: фамилия, имя, отчество, логин, пароль. (Для подтверждения личности учителя администратору требуется ввести свой логин и пароль.)

5. Вход преподавателей в информационную систему и разрешение на:

5.1. Просмотр результатов проведенного теста (с учетом одного или нескольких критериев одновременно):

5.1.1. По дате.

5.1.2. По конкретному учащемуся.

5.1.3. По конкретной группе.

5.1.4. По конкретной теме, конкретной дисциплины.

5.2. Получение ведомости/отчета по результатам тестирования конкретной группы по конкретной дисциплине.

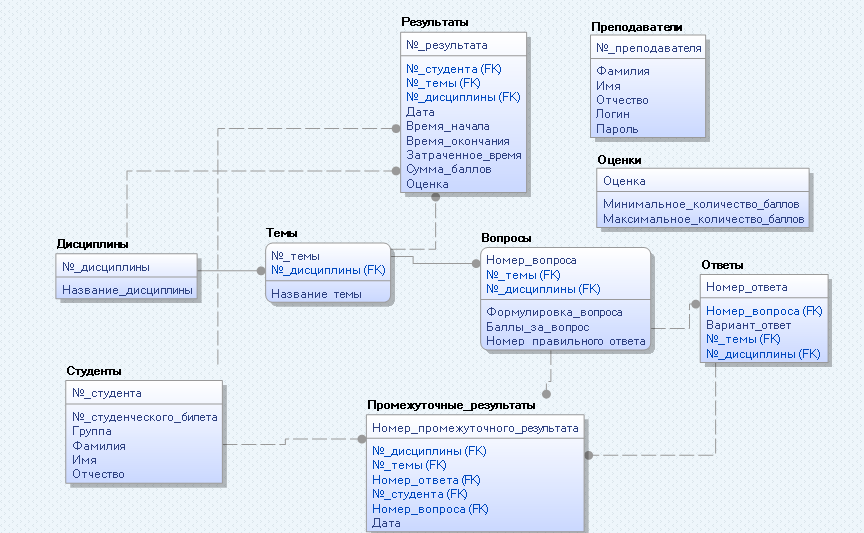
5.3. Получение отчета по итогам теста (с перечнем вопросов и ответов) для решения спорных вопросов.

5.4. Очистку журнала регистрации студентов (всех или выбранных).

5.5. Ввод новых дисциплин, тем, вопросов, и ответов по ним.

5.6. Установление и изменение критериев оценки с автоматическим пересчетом и заменой баллов за вопросы.

# **Логическая схема базы данных.**



# **Описание таблиц.**

**Студенты** (Номер\_студента, группа, фамилия, имя, отчество, номер\_студенческого\_билета)- предназначена для хранения данных об учащемся.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_студента | Int | Pk |
| Группа | Nvarchar(20) |  |
| Фамилия | Nvarchar(20) |  |
| Имя | Nvarchar(20) |  |
| Отчество | Nvarchar(20) |  |
| номер\_студенческого\_билета | Nvarchar(20) |  |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Группа, фамилия, имя, отчество, номер\_студенческого\_билета - прямо зависят от первичного ключа – Номер\_студента)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа *(Группа, фамилия, имя, отчество, номер\_студенческого\_билета - между собой независимы).*

**Дисциплины** (Номер\_дисциплины, название\_дисциплины)- предназначена для хранения дисциплин.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_дисциплины | Int | Pk |
| Название\_дисциплины | Nvarchar(50) |  |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Название\_дисциплины- прямо зависят от первичного ключа – Номер\_дисциплины)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа.

**Темы** (Номер\_темы, номер\_дисциплины, название\_темы)- предназначена для хранения тем.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_темы | Int | Pk |
| Название\_темы | Nvarchar(50) |  |
| Номер\_дисциплины | Int | Pk, Fk |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Название\_темы, Номер\_дисциплины - прямо зависят от первичного ключа –Номер\_темы, Номер\_дисциплины)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа.

**Учителя** (Фамилия, имя, отчество, номер\_учителя, логин, пароль)- предназначена для хранения данных о преподавателях.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Фамилия | Nvarchar(20) |  |
| Имя | Nvarchar(20) |  |
| Отчество | Nvarchar(20) |  |
| Номер\_учителя | Int | Pk |
| Логин | Nvarchar(20) |  |
| Пароль | Nvarchar(20) |  |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Фамилия, имя, отчество, логин, пароль - прямо зависят от первичного ключа – Номер\_учителя)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Фамилия, имя, отчество, логин, пароль - между собой независимы*)

**Результаты** (Номер\_студента, номер\_дисциплины, номер\_темы, Дата, Начало, Конец, Среднее\_время, Сумма\_баллов, Оценка, Номер\_результата)- предназначена для хранения данных о результатах тестирования.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_студента | Int | Fk |
| Номер\_дисциплины | Int | Fk |
| Номер\_темы | Int | Fk |
| Дата | Date |  |
| Начало | Time(0) |  |
| Конец | Time(0) |  |
| Среднее\_время | Int |  |
| Сумма\_баллов | Int |  |
| Оценка | Int |  |
| Номер\_результата | Int | Pk |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Номер\_студента, номер\_дисциплины, номер\_темы, Дата, Начало, Конец, Среднее\_время, Сумма\_баллов, Оценка - прямо зависят от первичного ключа – Номер\_результата)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Номер\_студента, номер\_дисциплины, номер\_темы, Дата, Начало, Конец, Среднее\_время, Сумма\_баллов, Оценка - между собой независимы*)

**Промежуточные\_результаты** (Номер\_ответа, номер\_темы, номер\_дисциплины, номер\_вопроса, номер\_студента, Дата, Номер\_промежуточного\_результата)- предназначена для хранения данных о промежуточных результатах тестирования.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_ответа | Int | Fk |
| Номер\_темы | Int | Fk |
| Номер\_дисциплины | Int | Fk |
| Номер\_вопроса | Int | Fk |
| Номер\_студента | Int | Fk |
| Дата | Date |  |
| Номер\_промежуточных\_результатов | Int | Pk |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Номер\_ответа, номер\_темы, номер\_дисциплины, номер\_вопроса, номер\_студента, Дата - прямо зависят от первичного ключа –* *Номер\_промежуточного\_результата)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Номер\_ответа, номер\_темы, номер\_дисциплины, номер\_вопроса, номер\_студента, Дата - между собой независимы*)

**Вопросы** (Формулировка, Баллы, Номер\_вопроса, Номер\_темы, Номер\_дисциплины, Номер\_правильного\_ответа)- предназначена для хранения данных о вопросах по заданной теме заданной дисциплины.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Формулировка | Nvarchar(200) |  |
| Баллы | Int |  |
| Номер\_вопроса | Int | Pk |
| Номер\_темы | Int | Pk, Fk |
| Номер\_дисциплины | Int | Pk, Fk |
| Номер\_правильного\_ответа | Int |  |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(*(*Формулировка, Баллы, Номер\_правильного\_ответа - прямо зависят от первичного ключа – Номер\_вопроса, Номер\_темы, Номер\_дисциплины)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Формулировка, Баллы, Номер\_правильного\_ответа - между собой независимы*)

**Оценки** (Оценка, Минимальное\_количество\_баллов, Максимальное\_количество\_баллов)- предназначена для хранения данных о критериях оценки.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Оценка | Int | Pk |
| Минимальное\_количество\_баллов | Int |  |
| Максимальное\_количество\_баллов | Int |  |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Минимальное\_количество\_баллов, Максимальное\_количество\_баллов - прямо зависят от первичного ключа – Оценка)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Минимальное\_количество\_баллов, Максимальное\_количество\_баллов - между собой независимы*)

**Ответы** (Номер\_ответа, Номер\_вопроса, Вариант\_ответа, Номер\_темы, Номер\_дисциплины)- предназначена для хранения данных об ответах на определенный вопрос.

|  |  |  |
| --- | --- | --- |
| Название поля | Тип | Ключ |
| Номер\_ответа | Int | Pk |
| Номер\_вопроса | Int | Fk |
| Вариант\_ответа | Nvarchar(200) |  |
| Номер\_темы | Int | Fk |
| Номер\_дисциплины | Int | Fk |

Таблица приведена к 1 нормальной форме, так как каждый её атрибут, в любом допустимом значении имеет только 1 значение

Таблица приведена ко 2 нормальной форме, так как она находится в первой нормальной форме и каждый не ключевой атрибут не приводимо зависит от первичного ключа *(Номер\_вопроса, Вариант\_ответа, Номер\_темы, Номер\_дисциплины - прямо зависят от первичного ключа – Номер\_ответа)*

Таблица приведена к 3 нормальной форме, так как она находится во второй нормальной форме и каждый не ключевой атрибут не находится в транзитивной функциональной зависимости от потенциального ключа. (*Номер\_вопроса, Вариант\_ответа, Номер\_темы, Номер\_дисциплины - между собой независимы*)

# **Другие объекты базы данных (запросы, отчёты, хранимые процедуры, триггеры)**

**Триггеры, контролирующие целостность данных:**

Use КурсовойПроект

if exists(select \* from sysobjects where type='TR' and name='Delete1')

drop trigger Delete1

go

CREATE TRIGGER Delete1

On Results for Delete

as

delete from Student where Student.NumberStudent in (select NumberStudent from deleted)

go

Use КурсовойПроект

if exists(select \* from sysobjects where type='TR' and name='Delete2')

drop trigger Delete2

go

CREATE TRIGGER Delete2

On UnResults for Delete

as

delete from Student where Student.NumberStudent in (select NumberStudent from deleted)

go

**Хранимые процедуры:**

Use КурсовойПроект

IF EXISTS (SELECT name FROM sysobjects

WHERE name = 'Промежуточная\_отчетность' AND type = 'P')

DROP PROCEDURE Промежуточная\_отчетность

go

create proc Промежуточная\_отчетность

@fam nvarchar(20),

@numberstudentticket nvarchar(20) as

SELECT fam, NumberStudentTicket, Formulation, UnResults.NumberAnswer, TrueNumberAnswer, ThemeName, NameDiscipline

FROM Student INNER JOIN

UnResults ON Student.NumberStudent = UnResults.NumberStudent INNER JOIN

Answers ON UnResults.NumberAnswer = Answers.NumberAnswer INNER JOIN

Question ON UnResults.NumberQuestion = Question.NumberQuestion AND UnResults.NumberTheme = Question.NumberTheme AND UnResults.NumberDiscipline = Question.NumberDiscipline AND

Answers.NumberQuestion = Question.NumberQuestion AND Answers.NumberTheme = Question.NumberTheme AND Answers.NumberDiscipline = Question.NumberDiscipline INNER JOIN

Discipline ON UnResults.NumberDiscipline = Discipline.NumberDiscipline INNER JOIN

Theme ON Question.NumberTheme = Theme.NumberTheme AND Question.NumberDiscipline = Theme.NumberDiscipline AND Discipline.NumberDiscipline = Theme.NumberDiscipline

Where fam=@fam and NumberStudentTicket=@numberstudentticket;

Use КурсовойПроект

if exists(select \* from sysobjects where name='Выборка\_оценки' and type='p')

drop proc Выборка\_оценки

go

create proc Выборка\_оценки

@ball int

as

Select Marks.Mark From Marks Where Marks.MinPoint<=@ball and Marks.MaxPoint>=@ball;

Use КурсовойПроект

if exists(select \* from sysobjects where name='Получение\_количества\_баллов\_за\_вопрос' and type='p')

drop proc Получение\_количества\_баллов\_за\_вопрос

go

create proc Получение\_количества\_баллов\_за\_вопрос

@NumberQuestion int,

@VariousAnswer nvarchar(200),

@Symbol nvarchar(1) OUTPUT,

@ball int OUTPUT

as

if exists (Select Question.Point

FROM Question

Where Question.TrueNumberAnswer=(Select Answers.NumberAnswer

FROM Answers

Where Answers.VariousAnswer=@VariousAnswer and Answers.NumberQuestion=@NumberQuestion))

begin

set @Symbol='Y'

set @ball= (Select Question.Point

FROM Question

Where Question.TrueNumberAnswer=(Select Answers.NumberAnswer

FROM Answers

Where Answers.VariousAnswer=@VariousAnswer and Answers.NumberQuestion=@NumberQuestion))

end

else

set @Symbol='N';

Use КурсовойПроект

if exists(select \* from sysobjects where name='CountAndRand1' and type='p')

drop proc CountAndRand1

go

create proc CountAndRand1

@themename nvarchar(50),

@Disciplinename nvarchar(50)

as

SELECT NumberQuestion

FROM Question INNER JOIN

Discipline ON Question.NumberDiscipline = Discipline.NumberDiscipline INNER JOIN

Theme ON Question.NumberTheme = Theme.NumberTheme AND Question.NumberDiscipline = Theme.NumberDiscipline AND Discipline.NumberDiscipline = Theme.NumberDiscipline

Where Theme.ThemeName=@themename and Discipline.NameDiscipline=@Disciplinename;

Use КурсовойПроект

if exists(select \* from sysobjects where name='CountAndRand' and type='p')

drop proc CountAndRand

go

create proc CountAndRand

@themename nvarchar(50),

@Disciplinename nvarchar(50)

as

SELECT Count(\*) as Rand1

FROM Question INNER JOIN

Discipline ON Question.NumberDiscipline = Discipline.NumberDiscipline INNER JOIN

Theme ON Question.NumberTheme = Theme.NumberTheme AND Question.NumberDiscipline = Theme.NumberDiscipline AND Discipline.NumberDiscipline = Theme.NumberDiscipline

Where Theme.ThemeName=@themename and Discipline.NameDiscipline=@Disciplinename;

Use КурсовойПроект

if exists(select \* from sysobjects where name='SelectAnswers' and type='p')

drop proc SelectAnswers

go

create proc SelectAnswers

@number int

as

SELECT Question.Formulation, Answers.VariousAnswer

FROM Question INNER JOIN

Answers ON Question.NumberQuestion = Answers.NumberQuestion AND Question.NumberTheme = Answers.NumberTheme AND Question.NumberDiscipline = Answers.NumberDiscipline

Where Question.NumberQuestion=Answers.NumberQuestion and Question.NumberQuestion=@number;

Go

Use КурсовойПроект

if exists(select \* from sysobjects where name='TransProvTeacher' and type='p')

drop proc TransProvTeacher

go

create proc TransProvTeacher

@login nvarchar(20),

@password nvarchar(20),

@char nvarchar(1) OUTPUT as

if exists (Select \* from Teacher Where Teacher.login=@login and Teacher.password=@password)

set @char='y'

else

set @char='n';

Use КурсовойПроект

if exists(select \* from sysobjects where name='TransProvStudent' and type='p')

drop proc TransProvStudent

go

create proc TransProvStudent

@fam nvarchar(20),

@NumberStudentTicket nvarchar(20),

@char nvarchar(1) OUTPUT as

if exists (Select \* from Student Where Student.fam=@fam and Student.NumberStudentTicket=@NumberStudentTicket)

set @char='y'

else

set @char='n';

Use КурсовойПроект

if exists(select \* from sysobjects where name='TransEnterTeacher' and type='p')

drop proc TransEnterTeacher

go

create proc TransEnterTeacher @name nvarchar(20), @fam nvarchar(20), @otch nvarchar(20),

@login nvarchar(20), @password nvarchar(20), @pr char(1) as

--начало транзакции

BEGIN TRAN T1EnterTeacher

Insert Into Teacher (name, fam, otch, NumberTeacher, Teacher.login, Teacher.password)

Values (@name, @fam, @otch, (Select Count(Teacher.NumberTeacher) From Teacher)+1 , @login, @password);

if @pr='Y'

-- фиксация изменений

COMMIT TRAN T1

Else

-- откат

ROLLBACK TRAN T1

Go

Use КурсовойПроект

if exists(select \* from sysobjects where name='Формирование\_оценки' and type='p')

drop proc Формирование\_оценки

go

create proc Формирование\_оценки

@num int

as

Update Marks Set

MaxPoint=ABS(@num)-(ABS((Mark-5))\*(ABS(@num)/5)),

MinPoint=ABS(@num)-(ABS((Mark-5))\*(ABS(@num)/5))-(ABS(@num)/5)-1

Update Marks Set

MinPoint=0

Where Mark=1

**Транзакции:**

Use КурсовойПроект

if exists(select \* from sysobjects where name='Добавление\_результатов ' and type='p')

drop proc Добавление\_результатов

go

create proc Добавление\_результатов

@NumberTheme nvarchar(50),

@NumberDiscipline nvarchar(50),

@StudentFam nvarchar(30),

@StudentTicket nvarchar(30),

@startSt time(0),

@FinishSt time(0),

@SumPoint int,

@pr nvarchar(1) as

--начало транзакции

BEGIN TRAN T1EnterResult

Insert Into Results (NumberStudent, NumberDiscipline, NumberTheme, DateSt, StartSt, FinishSt, Average, SumPoint, Mark, NumberResult)

Values ((Select Student.NumberStudent From Student Where Student.fam= @StudentFam and Student.NumberStudentTicket=@StudentTicket),

(Select Discipline.NumberDiscipline From Discipline Where Discipline.NameDiscipline=@NumberDiscipline),

(Select Theme.NumberTheme From Theme Where Theme.ThemeName=@NumberTheme),

GETDATE(),@startSt,@finishSt, DATEDIFF(mi,@startSt, @finishSt), @SumPoint,

(Select Marks.Mark From Marks Where Marks.MinPoint<=@SumPoint and Marks.MaxPoint>=@SumPoint),

(Select Count(Results.NumberResult) From Results)+1)

if @pr='Y'

-- фиксация изменений

COMMIT TRAN T1

Else

-- откат

ROLLBACK TRAN T1

Go

Use КурсовойПроект

if exists(select \* from sysobjects where name='Добавление\_промежуточных\_результатов ' and type='p')

drop proc Добавление\_промежуточных\_результатов

go

create proc Добавление\_промежуточных\_результатов

@NumberQuestion int,

@NumberTheme nvarchar(50),

@NumberDiscipline nvarchar(50),

@NumberAnswer nvarchar(200),

@NumberStudent nvarchar(30),

@pr nvarchar(1) as

--начало транзакции

BEGIN TRAN T1EnterMark

Insert Into UnResults (DateSt, NumberAnswer, NumberDiscipline, NumberOfUnResults,

NumberQuestion, NumberStudent, NumberTheme)

Values (GETDATE(), (SELECT Answers.NumberAnswer FROM Answers Where Answers.VariousAnswer=@NumberAnswer and Answers.NumberQuestion=@NumberQuestion),

(Select Discipline.NumberDiscipline From Discipline Where Discipline.NameDiscipline=@NumberDiscipline), (Select Count(UnResults.NumberOfUnResults)+1 From UnResults),

@NumberQuestion, (SELECT Student.NumberStudent FROM Student Where Student.fam=@NumberStudent),

(Select Theme.NumberTheme From Theme Where Theme.ThemeName=@NumberTheme))

if @pr='Y'

-- фиксация изменений

COMMIT TRAN T1

Else

-- откат

ROLLBACK TRAN T1

Go

Use КурсовойПроект

if exists(select \* from sysobjects where name='TransEnterStudent' and type='p')

drop proc TransEnterStudent

go

create proc TransEnterStudent @name nvarchar(20), @fam nvarchar(20), @otch nvarchar(20),

@GroupSt nvarchar(20), @NumberStudentTicket nvarchar(20), @pr char(1) as

--начало транзакции

BEGIN TRAN T1EnterStudent

Insert Into Student (Student.name, Student.fam, Student.otch, Student.GroupSt,

Student.NumberStudent, Student.NumberStudentTicket)

Values (@name, @fam, @otch, @GroupSt, (Select MAX(Student.NumberStudent) From Student)+1 , @NumberStudentTicket);

if @pr='Y'

-- фиксация изменений

COMMIT TRAN T1

Else

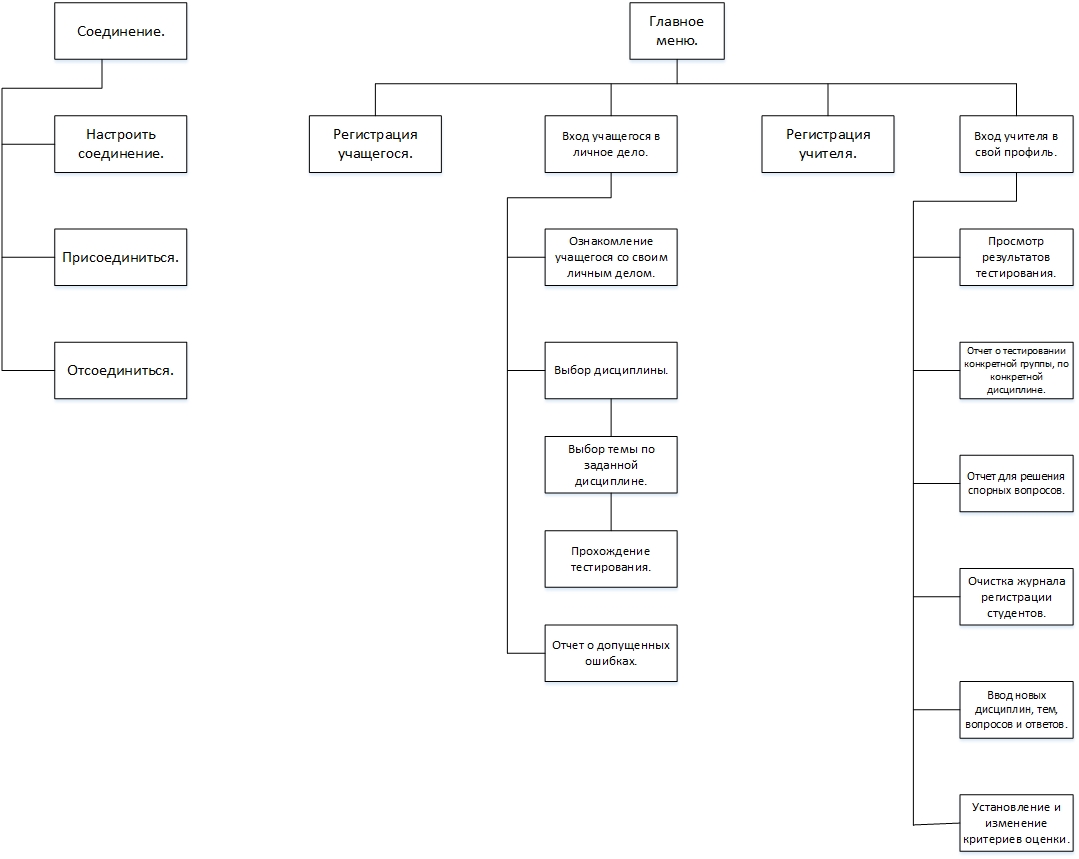
-- откат

ROLLBACK TRAN T1

Go

## **Разработка приложения**

## **Структурная схема приложения.**



# **Разработка интерфейса пользователя.**

Разработка интерфейса пользователя была основана на принципах эргономики и состоит из таких объектов как: "MainMenu", "Button", "label", "edit", "dbgrid", "dbedit" и другие.

Технология доступа к данным – ADO, используемые компоненты этой технологии:

«TADOQuery», «TADOConnection», «TADOStoredProc».

«TADOQuery» была использована в качестве основного набора данных, все вспомогательные функции осуществлялись через «TADOStroredProc».

Соответствие графических объектов:

TMainMenu – Основное меню выбора функций. (Главное меню программы)

TButton – Использовался как наглядный компонент выбора каких-либо вложенных функций.

TLabel – использовался как небольшие информативные поля. (Подсказки функций, подсказки определения вводимых полей)

TEdit – использовался для ввода полей, ввода фильтров.

TDBgrid – использовался для отображения набора данных из таблиц базы данных.

Для реализации отчётов, была использована технология "Fast Report", используемые компоненты:

TfrxReport - использовался для создания формы отчёта.

TfrxDBDataSet – использовался, как набор данных для отчёта.

# **Описание процесса отладки приложения.**

Отладка проводилась на основании возникновения ошибок при комплексном тестировании реализуемого функционала программы.

Во время отладки, сложно разрешимые, в первое время, проблемы возникали по вине различных событий, так как специфика отладки не позволяла выявить проблему в обработчиках событий, в связи с этим, затраты времени на поиск ошибок – увеличивались.

Примеры конкретных ошибок:

|  |  |
| --- | --- |
| Ошибка | Решение |
| Использование в запросе полей, пустого набора данных. | Проверка на пустоту набора данных, перед использованием его в запросах. |
| Различные семантические и синтаксические ошибки. | Исправление ошибок. |
| Ошибки в построении запросов. | Отладка запросов с помощью программы: "Microsoft SQL Server Management Studio" или ошибка опечаток в самой программе в компоненте TADOQuery. |
| Ошибки вывода данных в отчете. | Исправление выводимого набора данных в отчете. |
| Неправильная постановка задачи. (Генерация вопросов на рандом) | Пересмотренные проблемы и замена массива, использованного ранее, на списки. Причина неправильной постановки задачи: трудности при удалении последнего элемента массива. |
| Ошибки при работе со списком для генерации вопросов на рандом. (Ошибка обращения по адресу) | Нахождение и устранение неполадок при работе с памятью с помощью дампов памяти. |

# **Выбор способа защиты базы данных.**

База данных защищена уровнями доступа (авторизация):

1. Системный администратор (максимальный доступ к БД).
2. Учитель (доступ к некоторым таблицам БД).

Защита от ввода некорректных данных:

1. Проверка правильности текстовых полей.
2. Проверка правильности числовых полей.

Обеспечение целостности данных с помощью триггеров.

Внутренние функции шифрования Microsoft SQL Server 2008 R2.

# **Инструкция пользователю.**

# **Общие сведения об информационной системе.**

Информационная система «Программа контроля знаний учащихся» реализует функции направленные на:

1. Оптимизацию времени урока.
2. Сокращение труда учителей при проверке знаний учащихся.
3. Быструю и точную оценку результатов знаний учащихся.
4. Упрощение ознакомления преподавателя с результатами тестирования.

В следствии использования данной информационной системы в ученых заведениях существенно возрастет производительность труда преподавателей.

# **Требования к техническим средства.**

Компьютер, ноутбук, ультрабук или нетбук- стандартной комплектации с следующими минимальными требованиями:

1) Процессор: Intel Atom 1,4ГГц.

2) ОЗУ: 1 ГБ.

3) Видео память: 512 мегабайт.

4) Мышь.

5) Клавиатура.

6) Монитор.

## **Требования к программным средствам.**

1. MS SQL Server 2008 R2 и выше.

# **Настройка информационной системы.**

Настройка информационной системы осуществляется средствами технологии "ADO":

1) Настройка драйверов.

2) Настройка авторизации.

3) Выбор сервера.

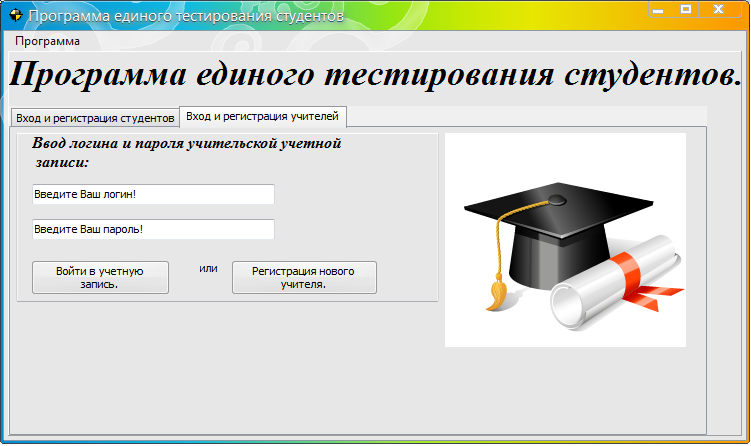
4) Выбор базы данных.

# **Формы ввода.**

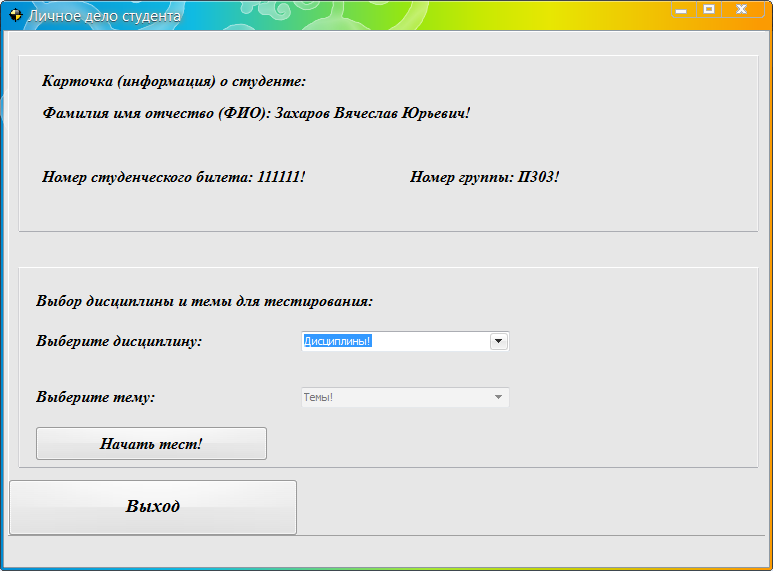
Разработка форм ввода была основана на принципах эргономики и информативности, в основе расположения компонентов лежит стремление к фрактальному стилю, что упрощает скорость набора данных, каждый фрактал состоит из части описания и части ввода.

Формы ввода:

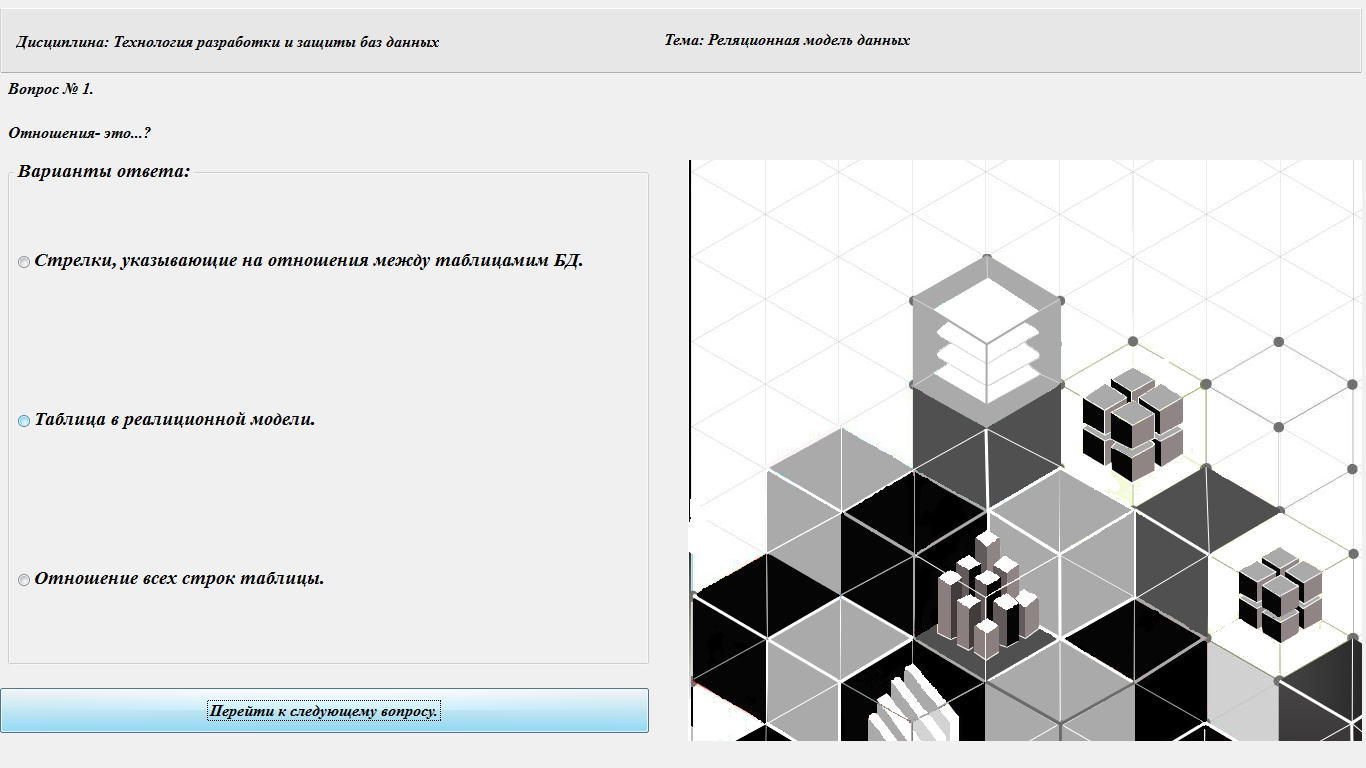
1. Начальная (главная) форма. Форма входа для преподавателей и студентов.



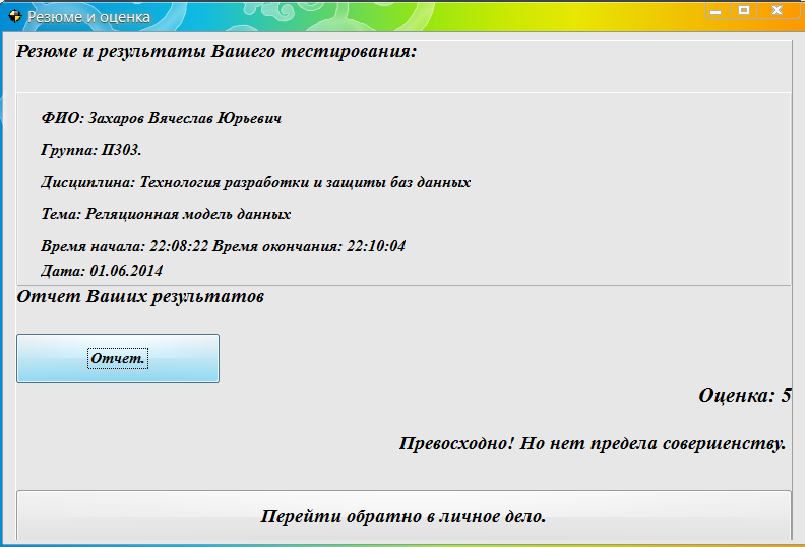
1. Форма выбора дисциплины и темы для прохождения темы.



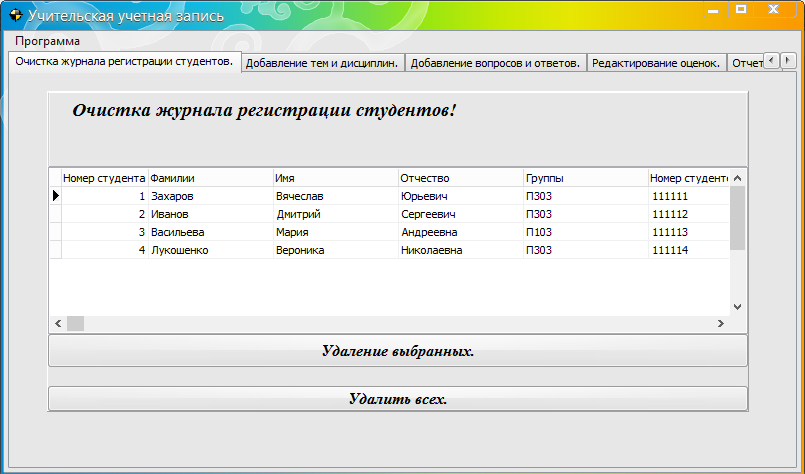
1. Форма тестирования.



1. Форма резюме.



1. Форма преподавателя.



# **Отчеты.**

Отчёты реализованы в соответствии с предъявленными к ним требованиями.

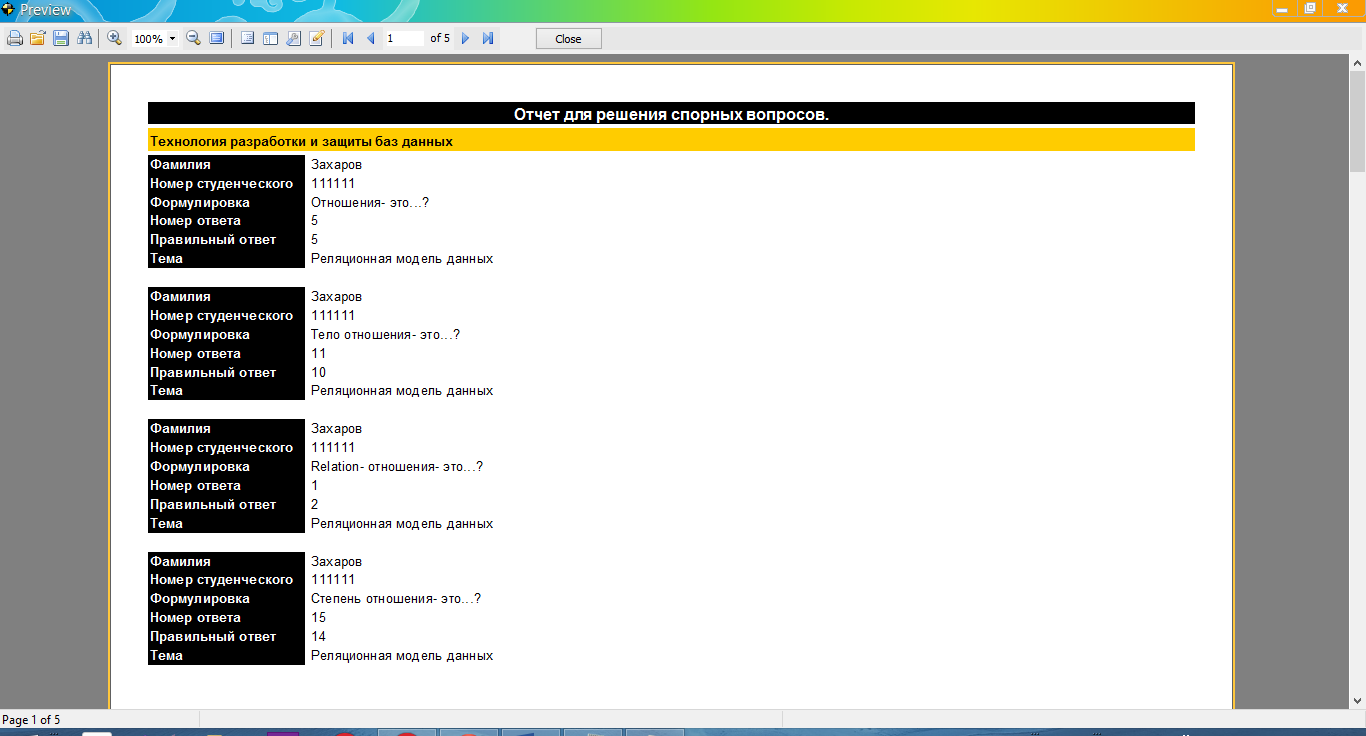
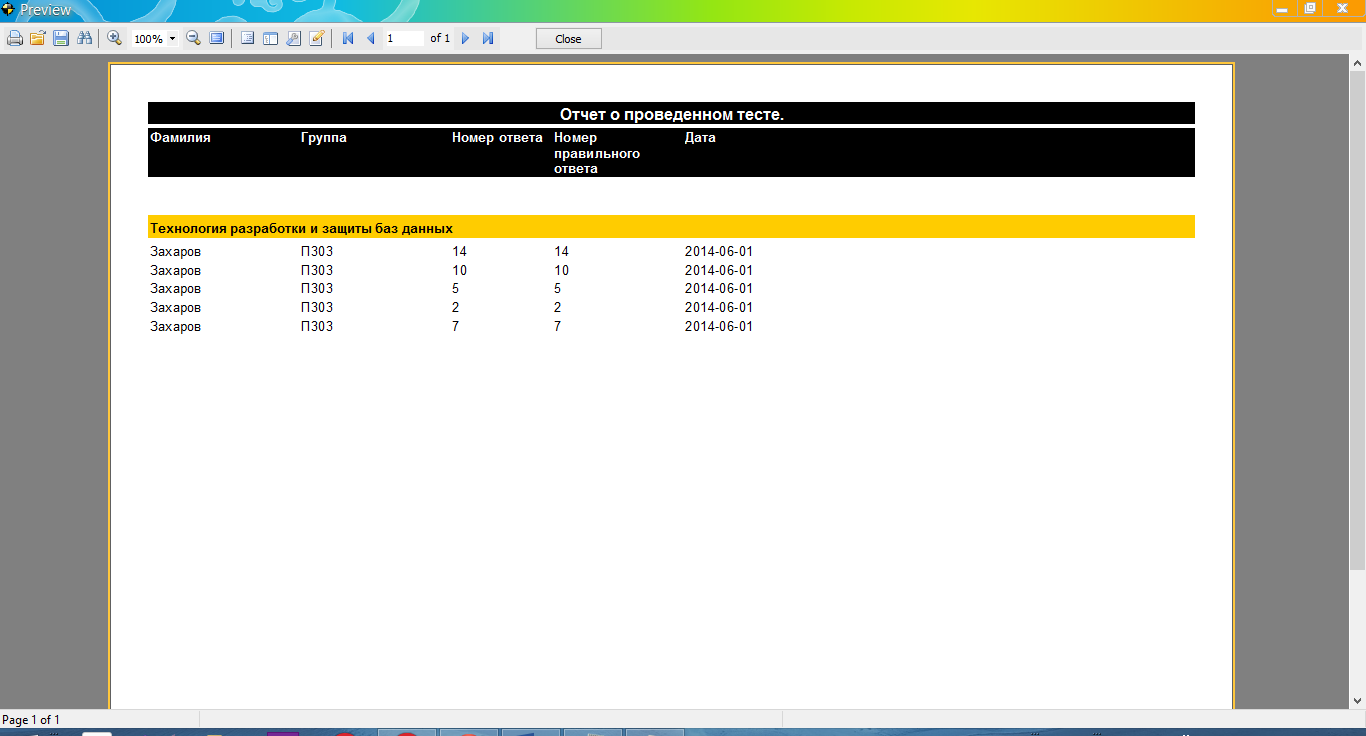
Отчеты содержат:

1. Отчет по результатам тестирования определенной группы по определенной дисциплине.

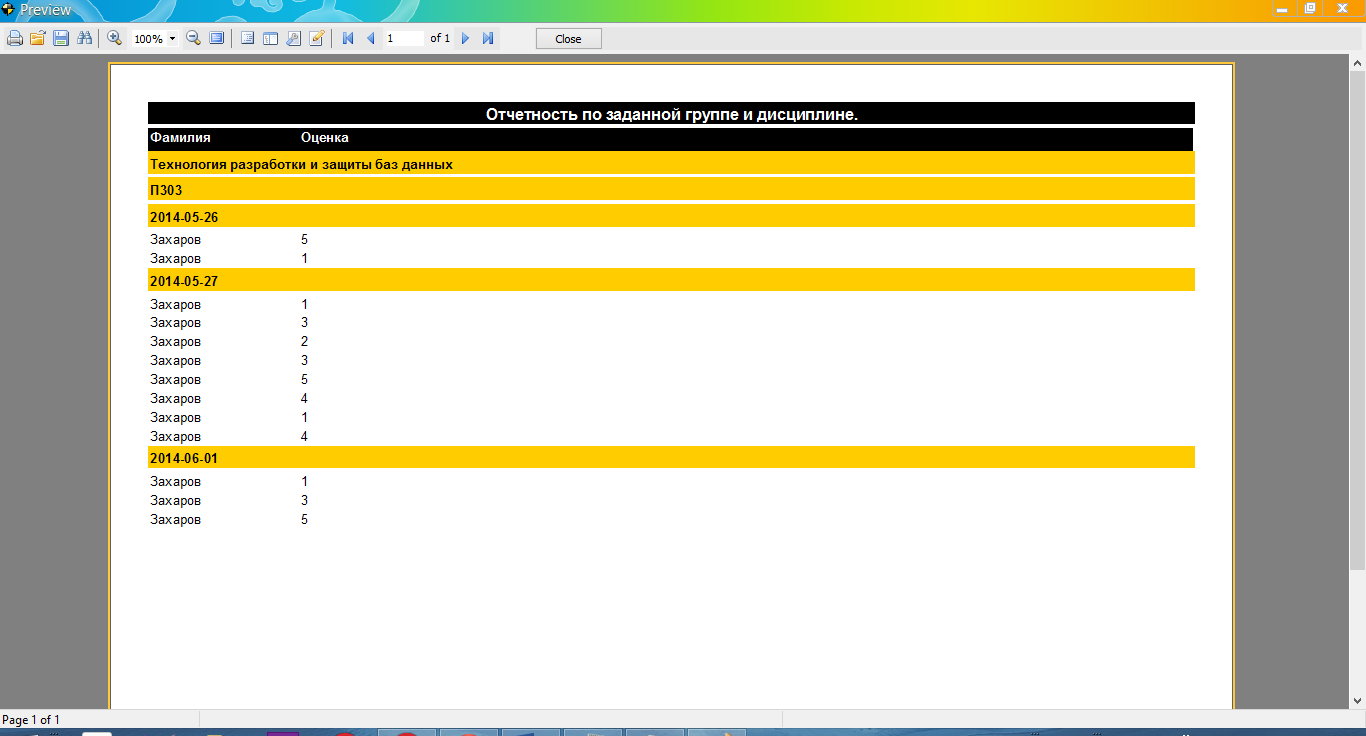
2. Отчет по итогам теста для решения спорных вопросов.

3. Отчет с ошибками учащегося допущенными при тестировании.

Отчеты:



# 



# **Защита информационной системы.**

Защита информационной системы в части целостности данных осуществляется средствами Microsoft SQL Server 2008 R2.

Защита от несанкционированного доступа осуществляется разграничением прав доступа к серверу баз данных.

Защита от некорректного доступа реализуется средствами приложения.

# **Спецификация.**

* 1. **Настройка соединения.**
     1. **Описание и приоритеты.**

Высокий приоритет. Данная функция позволяет настраивать соединение с базой данных (Название сервера, данные авторизации (логин, пароль), выбор драйверов и т. д.).

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии на пункт меню «Настройка соединения», появляется меню, позволяющее настроить параметры, описанные в пункте 3.1.1.

* + 1. **Функциональные требования.**

Реализовывается при помощи встроенной настройки TADOConnection в системе ADO.

* 1. **Присоединение.**
     1. **Описание и приоритеты.**

Высокий приоритет, данная функция позволяет выполнить подключение к серверу.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии на пункт меню «Подключение», выполняется подключение к серверу.

* + 1. **Функциональные требования.**

В случае неудачного подключения, вывести сообщение об этом и продолжить работу.

* 1. **Отсоединение.**
     1. **Описание и приоритеты.**

Низкий приоритет, данная функция позволяет выполнить отсоединение от сервера.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии на пункт меню «Отсоединение», выполняется отсоединение сервера.

* + 1. **Функциональные требования.**

Сначала закрыть открытые наборы данных, затем отключить соединение с сервером.

* 1. **Вход в личное дело.**
     1. **Описание и приоритеты.**

Высокий приоритет, данная функция входит в личное дело студента.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатие на кнопку «Войти в личное дело» учащийся может войти в свое личное дело при учете, что его данные были введены верно.

* + 1. **Функциональные требования.**

В случае правильно введённых данных- открыть форму с личным делом студента, в случае не правильно введенных данных- вывести ошибку.

* 1. **Прохождение тестирования.**
     1. **Описание и приоритеты.**

Высокий приоритет, данный модуль осуществляет функции прохождения тестирования учащимся.

* + 1. **Причинно-следственные связи, алгоритмы.**

После выбора учащимся в его личном деле дисциплины и темы по ней и нажатие кнопки «Пройти тестирование», учащийся проходит тестирование.

* + 1. **Функциональные требования.**

Чередование вопросов в произвольном порядке и исключение дублирования вопросов.

* 1. **Регистрация учащегося.**
     1. **Описание и приоритеты.**

Низкий приоритет. Данная функция вносит данные об учащемся в базу данных.

* + 1. **Причинно-следственные связи, алгоритмы.**

После нажатия кнопки «Регистрация» учащийся попадает в окно регистрации. После заполнения всех данных в этом окне и нажатие кнопки «Зарегистрироваться» личные данные учащегося вносятся в базу данных.

* + 1. **Функциональные требования.**

При успешной регистрации вывести сообщение об успехе. В обратном случае вывести причину неудачного внесения информации в базу данных.

* 1. **Вход в профиль преподавателя.**
     1. **Описание и приоритеты.**

Высокий приоритет. Данная функция входит в профиль учителя.

* + 1. **Причинно-следственные связи, алгоритмы.**

После нажатия кнопки войти в профиль, преподаватель может войти в свой профиль при учете верно введённых данных.

* + 1. **Функциональные требования.**

При верно введенных данных- отобразить профиль. При неверно введенных данных- вывести соответствующее сообщение.

* 1. **Просмотр результатов проведенного теста.**
     1. **Описание и приоритеты.**

Высокий приоритет. Данная функция позволяет просматривать результаты тестирования

* + 1. **Причинно-следственные связи, алгоритмы.**

После выборки критериев преподаватель может увидеть информацию о результатах по заданным критериям.

* + 1. **Функциональные требования.**

Отсутствуют.

* 1. **Очистка журнала регистрации.**
     1. **Описание и приоритеты.**

Средний приоритет. Данная функция позволяет очистить журнал регистрации студентов.

* + 1. **Причинно-следственные связи, алгоритмы.**

Преподаватель может выбрать кого удалить из журнала регистрации или удалить всех.

* + 1. **Функциональные требования.**

При успешном удалении выдать соответствующие сообщение. При неудачном- номер ошибки.

* 1. **Ввод новых дисциплин, тем, вопросов и ответов по ним.**
     1. **Описание и приоритеты.**

Средний приоритет. Данная функция реализует добавление информации в базу данных.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии кнопки добавить- осуществляется добавление в базу данных.

* + 1. **Функциональные требования.**

При успешном добавлении выдать сообщение об успешном добавлении, при не удачном- сообщение об ошибке.

* 1. **Изменение критериев оценки.**
     1. **Описание и приоритеты.**

Высокий приоритет. Данная функция позволяет изменить критерии оценивания теста.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии кнопки изменить- внести изменения на указанное в Edit число.

* + 1. **Функциональные требования.**

При успешном внесении изменений- выдать соответствующее сообщение, при наличие в Edit не числа или числа меньше 50- выдавать ошибку.

* 1. **Отчетность.**
     1. **Описание и приоритеты.**

Высокий приоритет. Данная функция выводит учительские отчеты.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии кнопки вывести отчет- должен выводиться отчет и отдельный модуль с данными для сортировки отчета.

* + 1. **Функциональные требования.**

Отсутствуют.

* 1. **Регистрация преподавателя.**
     1. **Описание и приоритеты.**

Приоритет низкий. Данная функция выполняет регистрацию в информационной системе нового преподавателя.

* + 1. **Причинно-следственные связи, алгоритмы.**

При нажатии кнопки регистрация- зарегистрировать, т. е. внести запись в базу данных о личных данных преподавателя.

* + 1. **Функциональные требования.**

Добавлять в БД данные о преподавателях разрешается только при соглашении на это администратора системы. При успешной регистрации выводить сообщение об успешной регистрации, при ошибке- номер ошибки.

# **Заключение.**

КП посвящён разработке информационной системы, предназначенной для автоматизации контроля знаний учащихся.

**В процессе работы были выполнены следующие пункты:**

1. Спроектирована база данных.

2. Разработано приложение для работы с базой данных.

3. Разработана спецификация и пояснительная записка к приложению.

4. Разработана презентация программного продукта.

**Возможности приложения:**

1. Возможность настройки соединения с сервером, драйверов и т.д.

2. Проведение тестирования.

2.1. Проведение тестирования по определенной теме, определенной дисциплины.

2.2. Ознакомление по результатам тестирования со своей оценкой, резюме и возможность просмотра отчета о допущенных ошибках.

3. Преподаватели.

* 1. Просмотр результатов проведенного теста (с учетом одного или нескольких критериев одновременно):
     1. По дате.
     2. По конкретному учащемуся.
     3. По конкретной группе.
     4. По конкретной теме, конкретной дисциплины.
  2. Получение ведомости/отчета по результатам тестирования конкретной группы по конкретной дисциплине.
  3. Получение отчета по итогам теста (с перечнем вопросов и ответов) для решения спорных вопросов.
  4. Очистку журнала регистрации студентов (всех или выбранных).
  5. Ввод новых дисциплин, тем, вопросов, и ответов по ним.
  6. Установление и изменение критериев оценки с автоматическим пересчетом и заменой баллов за вопросы.

4. Отчёты, исходя из требований:

* Отчет по результатам тестирования определенной группы по определенной дисциплине;
* Отчет для решения спорных вопросов по тестированию.

**В развитие информационной системы:**

1. Расширение системы до глобальной (единой государственной) системы тестирования.
2. Ввод новых видов вопросов. (аудио, видео, фото вопросы)
3. Расширение данных об учащихся в базе данных. (ввод новых полей и т.д.)
4. Создание нескольких уровней вопросов: легкие, средние, сложные, с соответствующим пересмотром баллов за вопрос.

# **Приложение. (Код программы)**

**Модуль1!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit15.h"

#include "Unit16.h"

#include "Unit17.h"

#include "Unit18.h"

#include "Unit6.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sSkinManager"

#pragma link "sButton"

#pragma link "sEdit"

#pragma link "sLabel"

#pragma link "sPageControl"

#pragma link "sPanel"

#pragma link "frxDCtrl"

#pragma link "frxADOComponents"

#pragma resource "\*.dfm"

#pragma resource "КП\_resources.RES"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

Form1->GridPanel1->Align=alClient;

/////////////////////////////

Form1->sLabel1->Caption="Ввод номера студенческого билета и фамилии студента:";

Form1->sEdit1->Text="Введите номер Вашего студенческого билета!";

Form1->sEdit2->Text="Введите Вашу фамилию!";

Form1->sButton1->Caption=("Войти в учетную запись.");

Form1->sButton2->Caption=("Регистрация нового студента.");

Form1->sLabel2->Caption=("или");

/////////////////////////////

Form1->sLabel3->Caption=("Ввод логина и пароля учительской учетной записи:");

Form1->sEdit3->Text=("Введите Ваш логин!");

Form1->sEdit4->Text=("Введите Вам пароль!");

Form1->sButton3->Caption=("Войти в учетную запись.");

Form1->sButton4->Caption=("Регистрация нового учителя.");

Form1->sLabel4->Caption=("или");

/////////////////////////////

Graphics::TBitmap \*ptBitmap;

ptBitmap=new Graphics::TBitmap();

TResourceStream\* ptRes =

new TResourceStream((int)HInstance, "Bitmap\_5",L"RT\_BITMAP");

ptBitmap->LoadFromStream(ptRes);

Image1->Picture->Assign((TPersistent\*)ptBitmap);

Image2->Picture->Assign((TPersistent\*)ptBitmap);

////////////////////////////////////////////////////////////////

Graphics::TBitmap \*ptBitmap1;

ptBitmap1=new Graphics::TBitmap();

TResourceStream\* ptRes1 =

new TResourceStream((int)HInstance, "Bitmap\_5",L"RT\_BITMAP");

ptBitmap1->LoadFromStream(ptRes1);

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sEdit1Click(TObject \*Sender)

{

Form1->sEdit1->Text="";

if (Form1->sEdit2->Text=="") {

Form1->sEdit2->Text="Введите Вашу фамилию!";

}

if (Form1->sEdit3->Text=="") {

Form1->sEdit3->Text="Введите Ваш логин!";

}

if (Form1->sEdit4->Text=="") {

Form1->sEdit4->Text="Введите Ваш пароль!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sEdit2Click(TObject \*Sender)

{

Form1->sEdit2->Text="";

if (Form1->sEdit1->Text=="") {

Form1->sEdit1->Text="Введите номер Вашего студенческого билета!";

}

if (Form1->sEdit3->Text=="") {

Form1->sEdit3->Text="Введите Ваш логин!";

}

if (Form1->sEdit4->Text=="") {

Form1->sEdit4->Text="Введите Ваш пароль!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sEdit3Click(TObject \*Sender)

{

Form1->sEdit3->Text="";

if (Form1->sEdit4->Text=="") {

Form1->sEdit4->Text="Введите Ваш пароль!";

}

if (Form1->sEdit1->Text=="") {

Form1->sEdit1->Text="Введите номер Вашего студенческого билета!";

}

if (Form1->sEdit2->Text=="") {

Form1->sEdit2->Text="Введите Вашу фамилию!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sEdit4Click(TObject \*Sender)

{

Form1->sEdit4->Text="";

if (Form1->sEdit3->Text=="") {

Form1->sEdit3->Text="Введите Ваш логин!";

}

if (Form1->sEdit2->Text=="") {

Form1->sEdit2->Text="Введите Вашу фамилию!";

}

if (Form1->sEdit1->Text=="") {

Form1->sEdit1->Text="Введите номер Вашего студенческого билета!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sButton1Click(TObject \*Sender)

{

if (DataModule1->ADOConnection1->Connected!=false)

{

//DataModule1->ADOStoredProc1->ProcedureName="TransProvStudent;1";

DataModule1->ADOStoredProc1->Parameters->ParamByName("@fam")->Value=(Form1->sEdit2->Text);

DataModule1->ADOStoredProc1->Parameters->ParamByName("@NumberStudentTicket")->Value=(Form1->sEdit1->Text);

DataModule1->ADOStoredProc1->ExecProc();

if(Form1->sEdit1->Text.IsEmpty())

{

ShowMessage("Вы не заполнили поле 'Номер студенческого билета'!");

}

else

if(Form1->sEdit2->Text.IsEmpty())

{

ShowMessage("Вы не заполнили поле 'Фамилия'!");

}

else

{

if (DataModule1->ADOStoredProc1->Parameters->ParamValues["@char"]=="y")

{

Form1->Hide();

Form2->ShowModal();

}

else

{

ShowMessage("Простите, но такого учащегося не существует!\n"

"Проверьте правильность ввода данных и зайдите снова.\n"

"Если Ваших данных ещё нет в системе- зарегистрируйтесь.");

}

}

}

else

{

ShowMessage("Вы не подключены к базе данных!\n"

"Для работы с программой выполните вход в базу данных \n"

"через главное меню программы.");

}

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sButton2Click(TObject \*Sender)

{

if (DataModule1->ADOConnection1->Connected!=false)

{

Form1->Hide();

Form3 = new TForm3(this);

Form3->ShowModal();

}

else

{

ShowMessage("Вы не подключены к базе данных!\n"

"Для работы с программой выполните вход в базу данных \n"

"через главное меню программы.");

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sButton3Click(TObject \*Sender)

{

if (DataModule1->ADOConnection1->Connected!=false)

{

DataModule1->ADOStoredProc2->Parameters->ParamByName("@login")->Value=(Form1->sEdit3->Text);

DataModule1->ADOStoredProc2->Parameters->ParamByName("@password")->Value=(Form1->sEdit4->Text);

DataModule1->ADOStoredProc2->ExecProc();

if(Form1->sEdit3->Text.IsEmpty())

{

ShowMessage("Вы не заполнили поле 'Логин'!");

}

else

if(Form1->sEdit4->Text.IsEmpty())

{

ShowMessage("Вы не заполнили поле 'Пароль'!");

}

else

{

if (DataModule1->ADOStoredProc2->Parameters->ParamValues["@char"]=="y")

{

Form1->Hide();

Form6->ShowModal(); //Вставить код новой формы.

}

else

{

ShowMessage("Простите, но такого преподавателя не существует!\n"

"Проверьте правильность ввода данных и зайдите снова.\n"

"Если Ваших данных ещё нет в системе- зарегистрируйтесь.");

}

}

}

else

{

ShowMessage("Вы не подключены к базе данных!\n"

"Для работы с программой выполните вход в базу данных \n"

"через главное меню программы.");

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::sButton4Click(TObject \*Sender)

{

if (DataModule1->ADOConnection1->Connected!=false)

{

Form1->Hide();

Form7 = new TForm7(this);

Form7->ShowModal();;

}

else

{

ShowMessage("Вы не подключены к базе данных!\n"

"Для работы с программой выполните вход в базу данных \n"

"через главное меню программы.");

}

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::N2Click(TObject \*Sender)

{

DataModule1->ADOConnection1->Connected=false;

AnsiString DS = "";

DS = PromptDataSource(reinterpret\_cast<unsigned int>(Handle), DataModule1->ADOConnection1->ConnectionString);

if (DS.Length() == 0)

{

AnsiString msg = "Вы отказались подключиться к базе дынных.";

MessageDlg(msg, mtInformation, TMsgDlgButtons() << mbNo << mbYes, 0);

}

else

{

AnsiString msg = "Подключение прошло успешно!";

ShowMessage(msg);

try {

DataModule1->ADOConnection1->ConnectionString= DS;

}

catch (...) {

}

}

DataModule1->ADOConnection1->Connected=true;

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::N3Click(TObject \*Sender)

{

DataModule1->ADOConnection1->Connected=false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::N4Click(TObject \*Sender)

{

Application->Terminate();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::FormClose(TObject \*Sender, TCloseAction &Action)

{

Application->Terminate();

}

//---------------------------------------------------------------------------

**Модуль 2!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit18.h"

#include "Unit16.h"

#include "Unit15.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sEdit"

#pragma link "sLabel"

#pragma resource "\*.dfm"

TForm3 \*Form3;

//---------------------------------------------------------------------------

\_\_fastcall TForm3::TForm3(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::FormActivate(TObject \*Sender)

{

Form3->GridPanel1->Align=alClient;

Form3->sEdit1->Text="Введите Вашу фамилию!";

Form3->sEdit2->Text="Введите Ваше имя!";

Form3->sEdit3->Text="Введите Ваше отчество!";

Form3->sEdit4->Text="Введите Вашу группу!";

Form3->sEdit5->Text="Введите Ваш номер студенческого билета!";

Form3->sLabel1->Caption="Ввод личной информации студента для регистрации:";

Form3->sButton1->Caption="Зарегестрировать студента.";

Form3->sButton2->Caption="Выход и откат изменений.";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sButton1Click(TObject \*Sender)

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Add("exec TransEnterStudent @fam=" + QuotedStr(Form3->sEdit1->Text)+

",@name=" + QuotedStr(Form3->sEdit2->Text)+

",@otch=" + QuotedStr(Form3->sEdit3->Text)+

",@GroupSt=" + QuotedStr(Form3->sEdit4->Text)+

",@NumberStudentTicket=" + QuotedStr(Form3->sEdit5->Text)+

",@pr='Y';");

DataModule1->ADOQuery1->ExecSQL();

ShowMessage("Вы успешно зарегестрировались!\n"

"Теперь Вы можете приступить к прохождению тестов!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sButton2Click(TObject \*Sender)

{

Form3->Hide();

Form3->Close();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::FormCreate(TObject \*Sender)

{

Form3->GridPanel1->Align=alClient;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sEdit1Click(TObject \*Sender)

{

Form3->sEdit1->Text=" ";

if (Form3->sEdit2->Text==" ") {

Form3->sEdit2->Text="Введите Ваше имя!";

}

if (Form3->sEdit3->Text==" ") {

Form3->sEdit3->Text="Введите Ваше отчество!";

}

if (Form3->sEdit4->Text==" ") {

Form3->sEdit4->Text="Введите Вашу группу!";

}

if (Form3->sEdit5->Text==" ") {

Form3->sEdit5->Text="Введите Ваш номер студенческого билета!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sEdit2Click(TObject \*Sender)

{

Form3->sEdit2->Text=" ";

if (Form3->sEdit1->Text==" ")

{

Form3->sEdit1->Text="Введите Вашу фамилию!";

}

if (Form3->sEdit3->Text==" ")

{

Form3->sEdit3->Text="Введите Ваше отчество!";

}

if (Form3->sEdit4->Text==" ")

{

Form3->sEdit4->Text="Введите Вашу группу!";

}

if (Form3->sEdit5->Text==" ")

{

Form3->sEdit5->Text="Введите Ваш номер студенческого билета!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sEdit3Click(TObject \*Sender)

{

Form3->sEdit3->Text=" ";

if (Form3->sEdit1->Text==" ")

{

Form3->sEdit1->Text="Введите Вашу фамилию!";

}

if (Form3->sEdit2->Text==" ")

{

Form3->sEdit2->Text="Введите Ваше имя!";

}

if (Form3->sEdit4->Text==" ")

{

Form3->sEdit4->Text="Введите Вашу группу!";

}

if (Form3->sEdit5->Text==" ")

{

Form3->sEdit5->Text="Введите Ваш номер студенческого билета!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sEdit4Click(TObject \*Sender)

{

Form3->sEdit4->Text=" ";

if (Form3->sEdit1->Text==" ")

{

Form3->sEdit1->Text="Введите Вашу фамилию!";

}

if (Form3->sEdit2->Text==" ")

{

Form3->sEdit2->Text="Введите Ваше имя!";

}

if (Form3->sEdit3->Text==" ")

{

Form3->sEdit3->Text="Введите Ваше отчество!";

}

if (Form3->sEdit5->Text==" ")

{

Form3->sEdit5->Text="Введите Ваш номер студенческого билета!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::sEdit5Click(TObject \*Sender)

{

Form3->sEdit5->Text=" ";

if (Form3->sEdit1->Text==" ")

{

Form3->sEdit1->Text="Введите Вашу фамилию!";

}

if (Form3->sEdit2->Text==" ")

{

Form3->sEdit2->Text="Введите Ваше имя!";

}

if (Form3->sEdit3->Text==" ")

{

Form3->sEdit3->Text="Введите Ваше отчество!";

}

if (Form3->sEdit4->Text==" ")

{

Form3->sEdit4->Text="Введите Вашу группу!";

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm3::FormClose(TObject \*Sender, TCloseAction &Action)

{

Form3->Hide();

Form3->Close();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

**Модуль 3!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit17.h"

#include "Unit15.h"

#include "Unit16.h"

#include "Unit4.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sComboBox"

#pragma resource "\*.dfm"

TForm2 \*Form2;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::sButton1Click(TObject \*Sender)

{

Form2->Hide();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormActivate(TObject \*Sender)

{

Form2->sComboBox2->Enabled=false;

Form2->sLabel4->Caption="Карточка (информация) о студенте:";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select \* From Student Where Student.fam=" + QuotedStr(Form1->sEdit2->Text) +

"and Student.NumberStudentTicket=" + QuotedStr(Form1->sEdit1->Text) + ";");

DataModule1->ADOQuery1->Open();

Form2->sLabel7->Caption="Фамилия имя отчество (ФИО): "+

DataModule1->ADOQuery1->FieldByName("fam")->AsString+ " "+

DataModule1->ADOQuery1->FieldByName("name")->AsString+ " " +

DataModule1->ADOQuery1->FieldByName("otch")->AsString+ "!";

Form2->sLabel8->Caption="Номер студенческого билета: "+

DataModule1->ADOQuery1->FieldByName("NumberStudentTicket")->AsString+"!";

Form2->sLabel9->Caption="Номер группы: "+

DataModule1->ADOQuery1->FieldByName("GroupSt")->AsString+ "!";

Form2->sLabel10->Caption="Выбор дисциплины и темы для тестирования:";

Form2->sLabel11->Caption="Выберите дисциплину:";

Form2->sLabel12->Caption="Выберите тему:";

Form2->sButton2->Caption="Начать тест!";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select \* From Discipline;");

DataModule1->ADOQuery1->Open();

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form2->sComboBox1->Items->Add(DataModule1->ADOQuery1->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery1->Next();

}

if(DataModule1->ADOQuery1->IsEmpty())

{

Form2->sComboBox1->Items->Add("Дисциплины отсутствуют!");

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::sComboBox1Change(TObject \*Sender)

{

Form2->sComboBox2->Enabled=true;

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select \* From Theme Inner Join "

" Discipline on Theme.NumberDiscipline= Discipline.NumberDiscipline"

" Where NameDiscipline="+QuotedStr(Form2->sComboBox1->Text)+";");

DataModule1->ADOQuery1->Open();

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form2->sComboBox2->Items->Add(DataModule1->ADOQuery1->FieldByName("ThemeName")->AsString);

DataModule1->ADOQuery1->Next();

}

if(DataModule1->ADOQuery1->IsEmpty())

{

Form2->sComboBox1->Items->Add("Темы отсутствуют!");

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::sButton2Click(TObject \*Sender)

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec CountAndRand @themename="+

QuotedStr(Form2->sComboBox2->Text)+", @Disciplinename="+

QuotedStr(Form2->sComboBox1->Text)+";");

DataModule1->ADOQuery1->Open();

if(DataModule1->ADOQuery1->IsEmpty())

{

ShowMessage("Простите, но вопросы по заданной теме пока отсутствуют! \n"

"Пройдите другой тест.");

}

else

{

Form2->Hide();

Form4->Close();

Form4 = new TForm4(this);

Form4->ShowModal();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormClose(TObject \*Sender, TCloseAction &Action)

{

if (MessageDlg("Вы уверены, что хотите выйти из программы?", mtConfirmation,TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes)

{

Application->Terminate();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::FormCreate(TObject \*Sender)

{

Form2->GridPanel1->Align=alClient;

}

//---------------------------------------------------------------------------

**Модуль4!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit4.h"

#include "Unit17.h"

#include "Unit16.h"

#include "Unit15.h"

#include "Unit5.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sGroupBox"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sGroupBox"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sGroupBox"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma resource "\*.dfm"

TForm4 \*Form4;

//---------------------------------------------------------------------------

struct Question {

int number;

struct Question \*next;

};

int i=0;

int countall=0;

int rand1=0;

int rand2;

TDateTime myTime;

//int \*mas;

int ball;

String sim1;

//int \*masbuff;

//int kol\_vo\_buff=0;

String timestart;

struct Question \*elem; /\* новый элемент \*/

struct Question \*start=NULL; /\* начало списка \*/

struct Question \*last=NULL;

struct Question \*prev=NULL;

//------------------------------------------------------------------------------

void

Del\_list (struct Question \*start)

{

while (start!=NULL) //Пока по адресу не пусто

{

Question \*temp=start->next; //Временная переменная для хранения адреса следующего элемента

delete start; //Освобождаем адрес обозначающий начало

start=temp; //Меняем адрес на следующий

}

}

//------------------------------------------------------------------------------

void CreateSp (struct Question \*i, /\* новый элемент \*/

struct Question \*\*start, /\* начало списка \*/

struct Question \*\*last) /\* конец списка \*/

{

struct Question \*old, \*p;

p = \*start;

if(!\*last)

{ /\* первый элемент в списке \*/

i->next = NULL;

\*last = i;

\*start = i;

return;

}

old = NULL;

while(p)

{

if(p->number != i->number)

{

old = p;

p = p->next;

}

}

(\*last)->next = i; /\* вставка в конец \*/

i->next = NULL;

\*last = i;

}

//------------------------------------------------------------------------------

void display(struct Question \*start)

{

while(start) {

ShowMessage(IntToStr(start->number));

start = start->next;

}

}

//------------------------------------------------------------------------------

int randomnum(

struct Question \*start, /\* начало списка \*/

int count)

{

int \*mas;

int rand1;

mas=(int\*)malloc(count \* sizeof(int));

for (int z=0; z!= count; z++)

{

mas[z]=start->number;

start = start->next;

}

Randomize();

rand1=mas[random(count)];

//ShowMessage(IntToStr(rand1));

return rand1;

delete []mas;

}

//--------------------------------------------------------------------------

void deletesp(

struct Question \*p, /\* предыдущий элемент \*/

struct Question \*i, /\* удаляемый элемент \*/

struct Question \*\*start, /\* начало списка \*/

struct Question \*\*last) /\* конец списка \*/

{

if(p) p->next = i->next;

else \*start = i->next;

if(i==\*last && p) \*last = p;

}

//---------------------------------------------------------------------------

\_\_fastcall TForm4::TForm4(TComponent\* Owner)

: TForm(Owner)

{

srand (time (NULL));

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::sButton1Click(TObject \*Sender)

{

struct Question \*prev=NULL;

sim1="Y";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec Добавление\_промежуточных\_результатов "

"@NumberAnswer="+ QuotedStr(Form4->sRadioGroup1->Items->Strings[Form4->sRadioGroup1->ItemIndex])+

" , @NumberDiscipline="+ QuotedStr(Form2->sComboBox1->Text)+

", @NumberQuestion="+ rand1 +" , @NumberStudent= "

+ QuotedStr(Form1->sEdit2->Text)+

" , @NumberTheme="+ QuotedStr(Form2->sComboBox2->Text)+" ,@pr= "+ QuotedStr(sim1)+";");

DataModule1->ADOQuery1->ExecSQL();

/\*DataModule1->ADOStoredProc4->Parameters->ParamByName("@NumberAnswer")->Value=

(Form4->sRadioGroup1->Items->Strings[Form4->sRadioGroup1->ItemIndex]);

DataModule1->ADOStoredProc4->Parameters->ParamByName("@NumberDiscipline")->Value=

(Form2->sComboBox1->Text);

DataModule1->ADOStoredProc4->Parameters->ParamByName("@NumberQuestion")->Value=

(rand1);

DataModule1->ADOStoredProc4->Parameters->ParamByName("@NumberStudent")->Value=

(Form1->sEdit2->Text);

DataModule1->ADOStoredProc4->Parameters->ParamByName("@NumberTheme")->Value=

(Form2->sComboBox2->Text);

DataModule1->ADOStoredProc4->ExecProc();\*/

DataModule1->ADOStoredProc3->Parameters->ParamByName("@VariousAnswer")->Value=

(Form4->sRadioGroup1->Items->Strings[Form4->sRadioGroup1->ItemIndex]);

DataModule1->ADOStoredProc3->Parameters->ParamByName("@NumberQuestion")->Value=rand1;

DataModule1->ADOStoredProc3->ExecProc();

if (DataModule1->ADOStoredProc3->Parameters->ParamValues["@Symbol"]=="Y")

{

ball=ball+DataModule1->ADOStoredProc3->Parameters->ParamByName("@ball")->Value;

//ShowMessage(IntToStr(ball));

}

if (countall!=0)

{

i++;

Form4->sLabel5->Caption="Вопрос № "+IntToStr(i)+".";

//Randomize();

//rand1=mas[random(countall)];

//ShowMessage(IntToStr(rand1));

rand1=randomnum(start,countall);

for (int r=0;r!=countall;r++)

{

/\*if(mas[r]==rand1)

{

for(int j=r;j<countall-1;j++)

{

mas[j]=mas[j+1];

}

countall--;

}\*/

/\*kol\_vo\_buff=0;

masbuff=(int\*)malloc((countall-1)\*(sizeof(int)));

for (int k=0; k < countall; k++)

{

if (mas[k]!=rand1)

{

kol\_vo\_buff++;

masbuff[kol\_vo\_buff]==mas[k];

}

} \*/

struct Question \*p;

p=start;

while(p)

{

if(rand1 == p->number)

deletesp(prev, p, &start, &last);

prev=p;

p = p->next;

}

}

//display(start);

countall--;

//delete []mas;

//mas=masbuff;

//delete []masbuff;

//for (int r=0;r!=countall;r++)

//{

//ShowMessage(IntToStr(mas[r]));

//}

Form4->sRadioGroup1->Items->Clear();

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec SelectAnswers @number="+QuotedStr(IntToStr(rand1))+";");

DataModule1->ADOQuery1->Open();

Form4->sLabel3->Caption=DataModule1->ADOQuery1->FieldByName("Formulation")->AsString;

for (int j=0; j < DataModule1->ADOQuery1->RecordCount; j++)

{

Form4->sRadioGroup1->Items->Add(DataModule1->ADOQuery1->FieldByName("VariousAnswer")->AsString);

DataModule1->ADOQuery1->Next();

}

}

else

{

i=0;

Del\_list(start);

start=NULL; /\* начало списка \*/

last=NULL;

prev=NULL;

countall=0;

rand1=0;

Form4->Hide();

Form5 = new TForm5(this);

Form5->ShowModal();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::FormShow(TObject \*Sender)

{

if (Form2->sComboBox1->Text=="Программирование на Delphi") {

Graphics::TBitmap \*ptBitmap;

ptBitmap=new Graphics::TBitmap();

TResourceStream\* ptRes =

new TResourceStream((int)HInstance, "Bitmap\_1",L"RT\_BITMAP");

ptBitmap->LoadFromStream(ptRes);

Image1->Picture->Assign((TPersistent\*)ptBitmap);

////////////////////////////////////////////////////////////////

Graphics::TBitmap \*ptBitmap1;

ptBitmap1=new Graphics::TBitmap();

TResourceStream\* ptRes1 =

new TResourceStream((int)HInstance, "Bitmap\_1",L"RT\_BITMAP");

ptBitmap1->LoadFromStream(ptRes1);

}

if (Form2->sComboBox1->Text=="Основы Access") {

Graphics::TBitmap \*ptBitmap;

ptBitmap=new Graphics::TBitmap();

TResourceStream\* ptRes =

new TResourceStream((int)HInstance, "Bitmap\_2",L"RT\_BITMAP");

ptBitmap->LoadFromStream(ptRes);

Image1->Picture->Assign((TPersistent\*)ptBitmap);

////////////////////////////////////////////////////////////////

Graphics::TBitmap \*ptBitmap1;

ptBitmap1=new Graphics::TBitmap();

TResourceStream\* ptRes1 =

new TResourceStream((int)HInstance, "Bitmap\_2",L"RT\_BITMAP");

ptBitmap1->LoadFromStream(ptRes1);

}

if (Form2->sComboBox1->Text=="Основы C++") {

Graphics::TBitmap \*ptBitmap;

ptBitmap=new Graphics::TBitmap();

TResourceStream\* ptRes =

new TResourceStream((int)HInstance, "Bitmap\_3",L"RT\_BITMAP");

ptBitmap->LoadFromStream(ptRes);

Image1->Picture->Assign((TPersistent\*)ptBitmap);

////////////////////////////////////////////////////////////////

Graphics::TBitmap \*ptBitmap1;

ptBitmap1=new Graphics::TBitmap();

TResourceStream\* ptRes1 =

new TResourceStream((int)HInstance, "Bitmap\_3",L"RT\_BITMAP");

ptBitmap1->LoadFromStream(ptRes1);

}

if (Form2->sComboBox1->Text=="Технология разработки и защиты баз данных") {

Graphics::TBitmap \*ptBitmap;

ptBitmap=new Graphics::TBitmap();

TResourceStream\* ptRes =

new TResourceStream((int)HInstance, "Bitmap\_4",L"RT\_BITMAP");

ptBitmap->LoadFromStream(ptRes);

Image1->Picture->Assign((TPersistent\*)ptBitmap);

////////////////////////////////////////////////////////////////

Graphics::TBitmap \*ptBitmap1;

ptBitmap1=new Graphics::TBitmap();

TResourceStream\* ptRes1 =

new TResourceStream((int)HInstance, "Bitmap\_4",L"RT\_BITMAP");

ptBitmap1->LoadFromStream(ptRes1);

}

static bool FullScreen = false;

static TRect Coord;

FullScreen = !FullScreen;

if (FullScreen) {

Coord = TRect(Left,Top,Width,Height);

BorderStyle = bsNone;

SetBounds(0,0,Screen->Width,Screen->Height);

} else {

BorderStyle = bsSizeable;

SetBounds(Coord.Left,Coord.Top,Coord.Right,Coord.Bottom);

}

timestart=FormatDateTime("hh:mm:ss", myTime.CurrentDateTime());

i++;

struct Question \*prev=NULL;

Form4->sLabel1->Caption="Дисциплина: "+Form2->sComboBox1->Text;

Form4->sLabel2->Caption="Тема: "+Form2->sComboBox2->Text;

Form4->sLabel5->Caption="Вопрос № "+IntToStr(i)+".";

Form4->sButton1->Caption="Перейти к следующему вопросу.";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec CountAndRand @themename="+

QuotedStr(Form2->sComboBox2->Text)+", @Disciplinename="+

QuotedStr(Form2->sComboBox1->Text)+";");

DataModule1->ADOQuery1->Open();

countall=DataModule1->ADOQuery1->FieldByName("Rand1")->AsInteger;

rand2=countall;

//ShowMessage(countall);

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec CountAndRand1 @themename="+

QuotedStr(Form2->sComboBox2->Text)+", @Disciplinename="+

QuotedStr(Form2->sComboBox1->Text)+";");

DataModule1->ADOQuery1->Open();

//mas=(int\*)malloc(countall\*sizeof(int));

for (int j=0; j != countall; j++)

{

elem= new Question;

elem->number=DataModule1->ADOQuery1->FieldByName("NumberQuestion")->AsInteger;

elem->next=NULL;

CreateSp(elem, &start, &last);

//mas[j]= DataModule1->ADOQuery1->FieldByName("NumberQuestion")->AsInteger;

DataModule1->ADOQuery1->Next();

//ShowMessage(mas[j]);

}

//display(start);

//Randomize();

//rand1=mas[random(countall)];

//ShowMessage(IntToStr(rand1));

rand1=randomnum(start,countall);

for (int r=0;r!=countall;r++)

{

/\*if(mas[r]==rand1)

{

for(int j=r;j<countall-1;j++)

{

mas[j]=mas[j+1];

}

countall--;

}

kol\_vo\_buff=0;

masbuff=(int\*)malloc((countall-1)\*(sizeof(int)));

for (int k=0; k < countall; k++)

{

if (mas[k]!=rand1)

{

kol\_vo\_buff++;

masbuff[kol\_vo\_buff]==mas[k];

}

} \*/

struct Question \*p;

p=start;

while(p)

{

if(rand1 == p->number)

deletesp(prev, p, &start, &last);

prev=p;

p = p->next;

}

}

//display(start);

countall--;

//delete mas;

//mas=masbuff;

//delete masbuff;

//for (int r=0;r<countall;r++)

//{

//ShowMessage(IntToStr(mas[r]));

//}

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Text=("exec SelectAnswers @number="+QuotedStr(IntToStr(rand1))+";");

DataModule1->ADOQuery1->Open();

Form4->sLabel3->Caption=DataModule1->ADOQuery1->FieldByName("Formulation")->AsString;

for (int j=0; j < DataModule1->ADOQuery1->RecordCount; j++)

{

Form4->sRadioGroup1->Items->Add(DataModule1->ADOQuery1->FieldByName("VariousAnswer")->AsString);

DataModule1->ADOQuery1->Next();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm4::FormClose(TObject \*Sender, TCloseAction &Action)

{

start=NULL; /\* начало списка \*/

last=NULL;

prev=NULL;

Del\_list(start);

}

//---------------------------------------------------------------------------

**Модуль 5!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit5.h"

#include "Unit4.h"

#include "Unit16.h"

#include "Unit15.h"

#include "Unit17.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "frxClass"

#pragma resource "\*.dfm"

TForm5 \*Form5;

extern String timestart;

extern int ball;

extern int rand2;

String sim2;

String timefinish;

TDateTime myTime;

//---------------------------------------------------------------------------

\_\_fastcall TForm5::TForm5(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::FormActivate(TObject \*Sender)

{

Form5->sButton2->Caption="Перейти обратно в личное дело.";

timefinish=FormatDateTime("hh:mm:ss", myTime.CurrentDateTime());

Form5->sLabel10->Caption="Резюме и результаты Вашего тестирования:";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select fam, name, otch, GroupSt "

"From Student Where NumberStudentTicket="+

QuotedStr(Form1->sEdit1->Text)+ ";");

DataModule1->ADOQuery1->Open();

Form5->sLabel1->Caption="ФИО: "+DataModule1->ADOQuery1->FieldByName("fam")->AsString+

" "+DataModule1->ADOQuery1->FieldByName("name")->AsString+" "+

DataModule1->ADOQuery1->FieldByName("otch")->AsString;

Form5->sLabel2->Caption="Группа: "+DataModule1->ADOQuery1->FieldByName("GroupSt")->AsString+".";

Form5->sLabel3->Caption="Дисциплина: "+Form2->sComboBox1->Text;

Form5->sLabel4->Caption="Тема: "+Form2->sComboBox2->Text;

Form5->sLabel5->Caption="Время начала: "+timestart+" Время окончания: "+timefinish;

Form5->sLabel6->Caption="Дата: "+Now().DateString();

Form5->sLabel7->Caption="Ваши баллы за тест: "+ IntToStr(ball);

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Add("exec Выборка\_оценки @ball="+ QuotedStr(IntToStr(ball))+ ";");

DataModule1->ADOQuery2->Open();

Form5->sLabel8->Caption="Оценка: "+DataModule1->ADOQuery2->FieldByName("mark")->AsString;

Form5->sLabel11->Caption="Отчет Ваших результатов";

Form5->sButton1->Caption="Отчет.";

switch (StrToInt(DataModule1->ADOQuery2->FieldByName("mark")->AsString))

{

case (1):

{

Form5->sLabel9->Caption="Ужасно! Вы плохо знаете тему. \n"

"Выучите материал и пересдайте тест. Удачи!";

break;

};

case (2):

{

Form5->sLabel9->Caption="Плохо. Вам нужно больше времени \n"

"уделять изучению материала! Удачи!";

break;

};

case (3):

{

Form5->sLabel9->Caption="Удовлетворительно, но можно и лучше! \n"

"Доучите тему и пересдайте. Удачи!";

break;

};

case (4):

{

Form5->sLabel9->Caption="Вы хорошо знаете материал. \n"

"Но все же не идеально... Удачи!";

break;

};

case (5):

{

Form5->sLabel9->Caption="Превосходно! Но нет предела совершенству. \n";

break;

};

}

sim2="Y";

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Text=("exec Добавление\_результатов"

" @NumberTheme="+ QuotedStr(Form2->sComboBox2->Text)+", @NumberDiscipline="+

QuotedStr(Form2->sComboBox1->Text)+", @StudentFam=" +

QuotedStr(DataModule1->ADOQuery1->FieldByName("fam")->AsString)+

", @StudentTicket="+ QuotedStr(Form1->sEdit1->Text)+

", @startSt="+ QuotedStr(timestart) +", @FinishSt="+QuotedStr(timefinish)+

", @SumPoint="+ QuotedStr(IntToStr(ball))+

", @pr="+QuotedStr(sim2)+";");

DataModule1->ADOQuery2->ExecSQL();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::sButton2Click(TObject \*Sender)

{

Form2->Close();

Form5->Close();

Form5->Hide();

Form2 = new TForm2(this);

Form2->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::sButton1Click(TObject \*Sender)

{

Form5->frxReport1->Variables->Variables["parametr1"]=rand2;

Form5->frxReport1->ShowReport();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm5::FormClose(TObject \*Sender, TCloseAction &Action)

{

if (MessageDlg("Вы уверены, что хотите выйти из программы?", mtConfirmation,TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes)

{

Application->Terminate();

}

}

//---------------------------------------------------------------------------

**Модуль 6!**

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit6.h"

#include "Unit16.h"

#include "Unit15.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sLabel"

#pragma link "sPanel"

#pragma link "sButton"

#pragma link "sComboBox"

#pragma link "sRadioButton"

#pragma link "sCheckBox"

#pragma link "sEdit"

#pragma link "sPageControl"

#pragma link "frxADOComponents"

#pragma link "frxClass"

#pragma link "frxDBSet"

#pragma resource "\*.dfm"

TForm6 \*Form6;

String cWhere;

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

\_\_fastcall TForm6::TForm6(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void ReopenQuery()

{

cWhere= "";

if (Form6->sComboBox1->Enabled==true)

{

cWhere = "fam=" + QuotedStr(Form6->sComboBox1->Text);

};

if (Form6->sComboBox2->Enabled==true)

{

if (cWhere!="")

{

cWhere= cWhere + " AND ";

}

cWhere="GroupSt="+ QuotedStr(Form6->sComboBox2->Text);

};

if (Form6->sComboBox3->Enabled==true)

{

if (cWhere!="")

{

cWhere = cWhere + " AND ";

}

cWhere="NameDiscipline="+ QuotedStr(Form6->sComboBox3->Text);

};

if (Form6->DateTimePicker1->Enabled==true)

{

if (cWhere!="")

{

cWhere = cWhere + " AND ";

}

cWhere="DateSt="+

QuotedStr(DateToStr(Form6->DateTimePicker1->Date.FormatString("dd.mm.yyyy")));

};

DataModule1->ADOQuery1->SQL->Clear();

if (cWhere!="")

{

DataModule1->ADOQuery1->SQL->Add("SELECT NumberResult as [Номер результата],"

"fam as [Фамилия], GroupSt as [Группа], ThemeName as [Название темы], "

"NameDiscipline as [Название дисциплины], DateSt as [Дата],"

"StartSt as [Время начала], "

" FinishSt as [Время конца], Average as [Затраченое время (мин.)], "

"SumPoint as [Сумма баллов], Mark as [Оценка]"

" FROM Results INNER JOIN"

" Student ON Results.NumberStudent = Student.NumberStudent INNER JOIN"

" Discipline ON Results.NumberDiscipline = Discipline.NumberDiscipline"

" INNER JOIN Theme ON Results.NumberTheme = Theme.NumberTheme AND "

"Results.NumberDiscipline = Theme.NumberDiscipline AND"

" Discipline.NumberDiscipline = Theme.NumberDiscipline Where "+cWhere+ ";");

}

else

{

DataModule1->ADOQuery1->SQL->Add("SELECT NumberResult as [Номер результата],"

"fam as [Фамилия], GroupSt as [Группа], ThemeName as [Название темы], "

"NameDiscipline as [Название дисциплины], DateSt as [Дата],"

"StartSt as [Время начала], "

" FinishSt as [Время конца], Average as [Затраченое время (мин.)], "

"SumPoint as [Сумма баллов], Mark as [Оценка]"

" FROM Results INNER JOIN"

" Student ON Results.NumberStudent = Student.NumberStudent INNER JOIN"

" Discipline ON Results.NumberDiscipline = Discipline.NumberDiscipline"

" INNER JOIN Theme ON Results.NumberTheme = Theme.NumberTheme AND "

"Results.NumberDiscipline = Theme.NumberDiscipline AND"

" Discipline.NumberDiscipline = Theme.NumberDiscipline");

}

DataModule1->ADOQuery1->Open();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::FormActivate(TObject \*Sender)

{

Form6->sButton6->Caption="Отчет по заданной группе и дисциплине.";

Form6->sButton4->Caption="Отчет по заданному студенту и числу.";

Form6->sComboBox8->Text="Выберите дисциплину!";

Form6->sComboBox9->Text="Выберите тему!";

Form6->sLabel4->Caption="Добавление дисциплины!";

Form6->sEdit1->Text="Введите название дисциплины!";

Form6->sButton3->Caption="Внести запись.";

Form6->sLabel5->Caption="Добавление темы!";

Form6->sEdit2->Text="Введите название темы!";

Form6->sComboBox7->Text="Выберите дисциплину!";

Form6->sButton5->Caption="Внести запись.";

Form6->sLabel6->Caption="Добавление ответа на вопрос!";

Form6->sEdit3->Text="Введите текст ответа!";

Form6->sButton7->Caption="Внести запись.";

Form6->sCheckBox5->Caption="Ответ правильный?";

Form6->sComboBox6->Text="Выберите вопрос к которому относится этот ответ.";

Form6->sLabel7->Caption="Добавление вопроса!";

Form6->sEdit4->Text="Введите текст вопроса!";

Form6->sButton10->Caption="Внести запись.";

Form6->sComboBox4->Text="Выберите дисциплину!";

Form6->sComboBox5->Text="Выберите тему!";

Form6->sLabel8->Caption="Изменение критериев оценки. Введите сумму баллов для оценки 5:";

Form6->sEdit5->Text="Введите число баллов!";

Form6->sButton8->Caption="Изменить критерий оценки";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select fam, name, otch from Teacher"

" Where login="+ QuotedStr(Form1->sEdit3->Text)+ " and password=" +

QuotedStr(Form1->sEdit4->Text)+";");

DataModule1->ADOQuery1->Open();

Form6->sLabel2->Caption="Добро пожаловать: "+

DataModule1->ADOQuery1->FieldByName("fam")->AsString+" "+

DataModule1->ADOQuery1->FieldByName("name")->AsString+" "+

DataModule1->ADOQuery1->FieldByName("otch")->AsString+"!";

Form6->sLabel1->Caption="Просмотр результатов тестирования: ";

Form6->sCheckBox1->Caption ="По дате.";

Form6->sCheckBox2->Caption="По учащемуся.";

Form6->sCheckBox3->Caption="По группе.";

Form6->sCheckBox4->Caption="По дисциплине.";

Form6->sComboBox1->Text="Фамилии!";

Form6->sComboBox2->Text="Группы!";

Form6->sComboBox3->Text="Дисциплины!";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select fam, GroupSt from Student;");

DataModule1->ADOQuery1->Open();

DataModule1->ADOQuery1->First();

//////////////////////////////////////////////////////////////////

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox1->Items->Add(DataModule1->ADOQuery1->FieldByName("fam")->AsString);

DataModule1->ADOQuery1->Next();

}

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox2->Items->Add(DataModule1->ADOQuery1->FieldByName("GroupSt")->AsString);

DataModule1->ADOQuery1->Next();

}

/////////////////////////////////////////////////////////////////Доработать

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("Select NameDiscipline From Discipline;");

DataModule1->ADOQuery1->Open();

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox3->Items->Add(DataModule1->ADOQuery1->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery1->Next();

}

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox7->Items->Add(DataModule1->ADOQuery1->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery1->Next();

}

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox4->Items->Add(DataModule1->ADOQuery1->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery1->Next();

}

DataModule1->ADOQuery1->First();

for (int i=0; i < DataModule1->ADOQuery1->RecordCount; i++)

{

Form6->sComboBox8->Items->Add(DataModule1->ADOQuery1->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery1->Next();

}

Form6->sLabel3->Caption="Очистка журнала регистрации студентов!";

Form6->sButton1->Caption="Удаление выбранных.";

Form6->sButton2->Caption="Удалить всех.";

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("SELECT NumberResult as [Номер результата],"

"fam as [Фамилия], GroupSt as [Группа], ThemeName as [Название темы], "

"NameDiscipline as [Название дисциплины], DateSt as [Дата],"

"StartSt as [Время начала], "

" FinishSt as [Время конца], Average as [Затраченое время (мин.)], "

"SumPoint as [Сумма баллов], Mark as [Оценка]"

" FROM Results INNER JOIN"

" Student ON Results.NumberStudent = Student.NumberStudent INNER JOIN"

" Discipline ON Results.NumberDiscipline = Discipline.NumberDiscipline"

" INNER JOIN Theme ON Results.NumberTheme = Theme.NumberTheme AND "

"Results.NumberDiscipline = Theme.NumberDiscipline AND"

" Discipline.NumberDiscipline = Theme.NumberDiscipline");

DataModule1->ADOQuery1->Open();

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Add("Select NumberStudent as [Номер студента], "

"fam as [Фамилии], name as [Имя], otch as [Отчество],"

"GroupSt as [Группы], NumberStudentTicket[Номер студентеческого]"

"From Student");

DataModule1->ADOQuery2->Open();

Form6->DBGrid2->Options<<dgMultiSelect;

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select Formulation From Question Where "

"NameDiscipline=" +QuotedStr(Form6->sComboBox8->Text)+ " and "+

"ThemeName="+ QuotedStr(Form6->sComboBox9->Text)+ ";");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sCheckBox1Click(TObject \*Sender)

{

if (Form6->sCheckBox1->Checked==true)

{

Form6->DateTimePicker1->Enabled=true;

ReopenQuery();

DataModule1->ADOQuery1->Filtered=true;

}

else

{

Form6->DateTimePicker1->Date=Date();

Form6->DateTimePicker1->Enabled=false;

ReopenQuery();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sCheckBox2Click(TObject \*Sender)

{

if (Form6->sCheckBox2->Checked==true)

{

Form6->sComboBox1->Enabled=true;

}

else

{

Form6->sComboBox1->Text="Фамилии!";

Form6->sComboBox1->Enabled=false;

ReopenQuery();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sCheckBox3Click(TObject \*Sender)

{

if (Form6->sCheckBox3->Checked==true)

{

Form6->sComboBox2->Enabled=true;

}

else

{

Form6->sComboBox2->Text="Группы!";

Form6->sComboBox2->Enabled=false;

ReopenQuery();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sCheckBox4Click(TObject \*Sender)

{

if (Form6->sCheckBox4->Checked==true)

{

Form6->sComboBox3->Enabled=true;

}

else

{

Form6->sComboBox3->Text="Дисциплины!";

Form6->sComboBox3->Enabled=false;

ReopenQuery();

}

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox1Change(TObject \*Sender)

{

ReopenQuery();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox2Change(TObject \*Sender)

{

ReopenQuery();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox3Change(TObject \*Sender)

{

ReopenQuery();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::DateTimePicker1Change(TObject \*Sender)

{

ReopenQuery();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton3Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select NameDiscipline From Discipline;");

DataModule1->ADOQuery3->Open();

if (DataModule1->ADOQuery3->IsEmpty()) {

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Insert Into Discipline (NumberDiscipline,"

" NameDiscipline) Values (1, "

+ QuotedStr(Form6->sEdit1->Text) + ");");

DataModule1->ADOQuery3->ExecSQL();

}

else

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Insert Into Discipline (NumberDiscipline,"

" NameDiscipline) Values ((Select MAX(NumberDiscipline) From Discipline)+1, "

+ QuotedStr(Form6->sEdit1->Text) + ");");

DataModule1->ADOQuery3->ExecSQL();

}

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select NameDiscipline From Discipline;");

DataModule1->ADOQuery3->Open();

sComboBox7->Items->Clear();

for (int i=0; i < DataModule1->ADOQuery3->RecordCount; i++)

{

Form6->sComboBox7->Items->Add(DataModule1->ADOQuery3->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery3->Next();

}

Form6->sEdit1->Text="";

ShowMessage("Дисциплина добавлена успешно!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton5Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select ThemeName From Theme"

" INNER JOIN Discipline ON Theme.NumberDiscipline = "

"Discipline.NumberDiscipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox7->Text)+ ";");

DataModule1->ADOQuery3->Open();

if (DataModule1->ADOQuery3->IsEmpty()) {

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Insert Into Theme (NumberDiscipline,"

" NumberTheme, ThemeName) Values ((Select NumberDiscipline From Discipline"

" where NameDiscipline="+ QuotedStr(Form6->sComboBox7->Text)+

"), 1, "+

QuotedStr(Form6->sEdit2->Text)+");");

DataModule1->ADOQuery3->ExecSQL();

}

else

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Insert Into Theme (NumberDiscipline,"

" NumberTheme, ThemeName) Values ((Select NumberDiscipline From Discipline"

" where NameDiscipline="+ QuotedStr(Form6->sComboBox7->Text)+

"), (Select MAX(NumberTheme) From Theme)+1, "+

QuotedStr(Form6->sEdit2->Text)+");");

DataModule1->ADOQuery3->ExecSQL();

}

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select NameDiscipline From Discipline;");

DataModule1->ADOQuery3->Open();

sComboBox4->Items->Clear();

for (int i=0; i < DataModule1->ADOQuery3->RecordCount; i++)

{

Form6->sComboBox4->Items->Add(DataModule1->ADOQuery3->FieldByName("NameDiscipline")->AsString);

DataModule1->ADOQuery3->Next();

}

Form6->sEdit2->Text="";

ShowMessage("Тема добавлена успешно!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox8Change(TObject \*Sender)

{

Form6->sComboBox9->Enabled=true;

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select ThemeName From Theme"

" INNER JOIN Discipline ON Theme.NumberDiscipline = "

"Discipline.NumberDiscipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox8->Text)+ ";");

DataModule1->ADOQuery3->Open();

sComboBox9->Items->Clear();

for (int i=0; i < DataModule1->ADOQuery3->RecordCount; i++)

{

Form6->sComboBox9->Items->Add(DataModule1->ADOQuery3->FieldByName("ThemeName")->AsString);

DataModule1->ADOQuery3->Next();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox4Change(TObject \*Sender)

{

Form6->sComboBox5->Enabled=true;

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Select ThemeName From Theme"

" INNER JOIN Discipline ON Theme.NumberDiscipline = "

"Discipline.NumberDiscipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox4->Text)+ ";");

DataModule1->ADOQuery3->Open();

sComboBox5->Items->Clear();

for (int i=0; i < DataModule1->ADOQuery3->RecordCount; i++)

{

Form6->sComboBox5->Items->Add(DataModule1->ADOQuery3->FieldByName("ThemeName")->AsString);

DataModule1->ADOQuery3->Next();

}

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sComboBox9Change(TObject \*Sender)

{

Form6->sComboBox6->Enabled=true;

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("SELECT Formulation"

" FROM Question INNER JOIN "

"Theme ON Question.NumberTheme = Theme.NumberTheme AND "

"Question.NumberDiscipline = Theme.NumberDiscipline INNER JOIN "

"Discipline ON Theme.NumberDiscipline = Discipline.NumberDiscipline "

"Where NameDiscipline="+ QuotedStr(Form6->sComboBox8->Text)+

" and ThemeName="+ QuotedStr(Form6->sComboBox9->Text)+";");

DataModule1->ADOQuery3->Open();

sComboBox6->Items->Clear();

for (int i=0; i < DataModule1->ADOQuery3->RecordCount; i++)

{

Form6->sComboBox6->Items->Add(DataModule1->ADOQuery3->FieldByName("Formulation")->AsString);

DataModule1->ADOQuery3->Next();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton10Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("SELECT Formulation"

" FROM Question INNER JOIN "

"Theme ON Question.NumberTheme = Theme.NumberTheme AND "

"Question.NumberDiscipline = Theme.NumberDiscipline INNER JOIN "

"Discipline ON Theme.NumberDiscipline = Discipline.NumberDiscipline "

"Where NameDiscipline="+ QuotedStr(Form6->sComboBox4->Text)+

" and ThemeName="+ QuotedStr(Form6->sComboBox5->Text)+";");

DataModule1->ADOQuery3->Open();

if (DataModule1->ADOQuery3->IsEmpty()) {

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Insert Into Question (Formulation,"

" NumberDiscipline, NumberQuestion, NumberTheme)"

"Values ("+ QuotedStr(Form6->sEdit4->Text)+ ", ("

"Select NumberDiscipline From Discipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox4->Text)+"), "

"1, (Select NumberTheme From Theme Where ThemeName="

+ QuotedStr(Form6->sComboBox5->Text)+"));");

DataModule1->ADOQuery3->ExecSQL();

}

else

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("Insert Into Question (Formulation,"

" NumberDiscipline, NumberQuestion, NumberTheme)"

"Values ("+ QuotedStr(Form6->sEdit4->Text)+ ", ("

"Select NumberDiscipline From Discipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox4->Text)+"), (Select Max(NumberQuestion) From "

"Question)+1, (Select NumberTheme From Theme Where ThemeName="

+ QuotedStr(Form6->sComboBox5->Text)+"));");

DataModule1->ADOQuery3->ExecSQL();

}

Form6->sEdit4->Text="";

ShowMessage("Вопрос добавлен успешно!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton1Click(TObject \*Sender)

{

if (MessageDlg("Вы уверены, что хотите удалить запись ?", mtConfirmation,TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes)

{

DataModule1->ADOQuery2->Delete();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton2Click(TObject \*Sender)

{

DataModule1->ADOQuery2->Close();

DataModule1->ADOQuery2->SQL->Clear();

DataModule1->ADOQuery2->SQL->Text="Delete From Student";

DataModule1->ADOQuery2->ExecSQL();

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton4Click(TObject \*Sender)

{

Form6->frxReport1->ShowReport();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton6Click(TObject \*Sender)

{

Form6->frxReport2->ShowReport();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton7Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Insert Into Answers (NumberAnswer, "

" NumberDiscipline, NumberQuestion, NumberTheme, VariousAnswer)"

" Values ((Select Count(NumberAnswer) From Answers)+1, "

"(Select NumberDiscipline From Discipline Where NameDiscipline="+

QuotedStr(Form6->sComboBox8->Text)+ "), (Select NumberQuestion From "

"Question Where Formulation="+ QuotedStr(Form6->sComboBox6->Text)+

"), (Select NumberTheme From Theme Where ThemeName=" +

QuotedStr(Form6->sComboBox9->Text)+"), "+ QuotedStr(Form6->sEdit3->Text)+

");");

DataModule1->ADOQuery3->ExecSQL();

if (Form6->sCheckBox5->Checked==true)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Update Question Set "

"TrueNumberAnswer=(Select Max(NumberAnswer) From Answers) "

"Where NumberQuestion=(Select NumberQuestion From "

"Question Where Formulation="+ QuotedStr(Form6->sComboBox6->Text)+");");

DataModule1->ADOQuery3->ExecSQL();

}

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Update Question Set"

" Point=(Select MaxPoint From Marks Where Mark='5')/"

" (Select Count(NumberQuestion) FROM Question INNER JOIN"

" Theme ON Question.NumberTheme = Theme.NumberTheme AND "

" Question.NumberDiscipline = Theme.NumberDiscipline INNER JOIN "

"Discipline ON Theme.NumberDiscipline = Discipline.NumberDiscipline "

" Where Question.NumberDiscipline=(Select NumberDiscipline From Discipline Where "

"NameDiscipline="+ QuotedStr(Form6->sComboBox8->Text)+ ") and "

"Question.NumberTheme=(Select NumberTheme From Theme Where ThemeName=" +

QuotedStr(Form6->sComboBox9->Text)+") Group By NameDiscipline, ThemeName);");

DataModule1->ADOQuery3->ExecSQL();

Form6->sEdit3->Text="";

ShowMessage("Ответ добавлен успешно!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sButton8Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("exec Формирование\_оценки "

"@num="+QuotedStr((Form6->sEdit5->Text)));

DataModule1->ADOQuery3->ExecSQL();

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Text=("Update Question Set"

" Point="+QuotedStr((Form6->sEdit5->Text))+"/5");

DataModule1->ADOQuery3->ExecSQL();

DataModule1->ADOQuery4->Close();

DataModule1->ADOQuery4->SQL->Clear();

DataModule1->ADOQuery4->SQL->Add("Select \* From Marks");

DataModule1->ADOQuery4->Open();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sTabSheet5Show(TObject \*Sender)

{

DataModule1->ADOQuery4->Close();

DataModule1->ADOQuery4->SQL->Clear();

DataModule1->ADOQuery4->SQL->Add("Select \* From Marks");

DataModule1->ADOQuery4->Open();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::FormClose(TObject \*Sender, TCloseAction &Action)

{

if (MessageDlg("Вы уверены, что хотите выйти из программы?", mtConfirmation,TMsgDlgButtons() << mbYes << mbNo, 0) == mrYes)

{

Application->Terminate();

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sTabSheet1Show(TObject \*Sender)

{

DataModule1->ADOQuery1->Close();

DataModule1->ADOQuery1->SQL->Clear();

DataModule1->ADOQuery1->SQL->Add("SELECT NumberResult as [Номер результата],"

"fam as [Фамилия], GroupSt as [Группа], ThemeName as [Название темы], "

"NameDiscipline as [Название дисциплины], DateSt as [Дата],"

"StartSt as [Время начала], "

" FinishSt as [Время конца], Average as [Затраченое время (мин.)], "

"SumPoint as [Сумма баллов], Mark as [Оценка]"

" FROM Results INNER JOIN"

" Student ON Results.NumberStudent = Student.NumberStudent INNER JOIN"

" Discipline ON Results.NumberDiscipline = Discipline.NumberDiscipline"

" INNER JOIN Theme ON Results.NumberTheme = Theme.NumberTheme AND "

"Results.NumberDiscipline = Theme.NumberDiscipline AND"

" Discipline.NumberDiscipline = Theme.NumberDiscipline");

DataModule1->ADOQuery1->Open();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::FormCreate(TObject \*Sender)

{

Form6->GridPanel1->Align=alClient;

Form6->GridPanel2->Align=alClient;

Form6->GridPanel3->Align=alClient;

Form6->GridPanel4->Align=alClient;

Form6->GridPanel5->Align=alClient;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N1Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=0;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N2Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=1;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N3Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=2;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N4Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=3;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N5Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=4;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N6Click(TObject \*Sender)

{

sPageControl1->ActivePageIndex=5;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::N7Click(TObject \*Sender)

{

Form6->Hide();

Form6->Close();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sEdit1Click(TObject \*Sender)

{

Form6->sEdit1->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sEdit2Click(TObject \*Sender)

{

Form6->sEdit2->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sEdit4Click(TObject \*Sender)

{

Form6->sEdit4->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm6::sEdit3Click(TObject \*Sender)

{

Form6->sEdit3->Text="";

}

//---------------------------------------------------------------------------

**Модуль 7!**

--------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit7.h"

#include "Unit16.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sEdit"

#pragma link "sLabel"

#pragma resource "\*.dfm"

TForm7 \*Form7;

//---------------------------------------------------------------------------

\_\_fastcall TForm7::TForm7(TComponent\* Owner)

: TForm(Owner)

{

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::FormActivate(TObject \*Sender)

{

Form7->sEdit1->Text="Введите Ваше имя!";

Form7->sEdit2->Text="Введите Вашу фамилию!";

Form7->sEdit3->Text="Введите Ваше отчество!";

Form7->sEdit4->Text="Введите Ваш логин!";

Form7->sEdit5->Text="Введите Ваш пароль!";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sEdit1Click(TObject \*Sender)

{

Form7->sEdit1->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sEdit2Change(TObject \*Sender)

{

Form7->sEdit2->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sEdit3Click(TObject \*Sender)

{

Form7->sEdit3->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sEdit4Click(TObject \*Sender)

{

Form7->sEdit4->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sEdit5Click(TObject \*Sender)

{

Form7->sEdit5->Text="";

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sButton1Click(TObject \*Sender)

{

DataModule1->ADOQuery3->Close();

DataModule1->ADOQuery3->SQL->Clear();

DataModule1->ADOQuery3->SQL->Add("exec TransEnterTeacher"

" @name="+ QuotedStr(Form7->sEdit1->Text)", @fam="

+ QuotedStr(Form7->sEdit2->Text)", @otch="

+ QuotedStr(Form7->sEdit3->Text)", @login="

+ QuotedStr(Form7->sEdit4->Text)", @password="

+ QuotedStr(Form7->sEdit3->Text)";");

DataModule1->ADOQuery3->ExecSQL();

ShowMessage("Преподаватель успешно добавлен!");

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::FormClick(TObject \*Sender)

{

Form7->GridPanel1->Align=AlClient;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::sButton2Click(TObject \*Sender)

{

Form7->Hide();

Form7->Close();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm7::FormClose(TObject \*Sender, TCloseAction &Action)

{

Form3->Hide();

Form3->Close();

Form1->ShowModal();

}

//---------------------------------------------------------------------------

# **Используемая литература:**

Справочник по среде RAD Studio Embarcadero XE5.

Справочник по основным библиотекам C++ Builder.

Программирование в C++ Builder, 7 изд, Год: 2010, Автор: Архангельский А.Я., Издательство: Бином.