**Приднестровский Государственный Университет**

**им. Т. Г. Шевченко**

# Кафедра АСОИиУ

#  Курсовая работа

**по дисциплине «Технология программирования»**

на тему:

Система рейтингого контроля знаний .

**Работу выполнили: Работу проверили:**

**студент ИТФ –96В преподаватели:**

**Кольцов Е.С. Галюшина О.В.**

## Тирасполь 1999 г.

## Содержание:

Задание на курсовую работу

1.1 Задание 4

1.2 Реализация 4

Введение

# Глава I:Системы поддержки принятия решений

1. Система Paradox

Глава II: Особенности объектно-ориентированного анализа 19

Информационная модель 21

Модель состояний для объекта «Операции над БД» 22

Модель состояний для объекта «Ввод в базу» 25

Модель взаимодействия объектов 26

Таблица переходов в состояния для объекта «Операции над БД» 27

Таблица переходов в состояния для объекта «Ввод в бузу» 28

Список событий 29

ДПДД для состояния «Инициализация» объекта «Операции над БД» 40

ДПДД для состояния «Создание БД» объекта «Операции над БД» 40

ДПДД для состояния «Открытие БД» объекта «Операции над БД» 41

ДПДД для состояния «Редактирование БД» объекта «Операции над БД» 41

ДПДД для состояния «Закрытие БД» объекта «Операции над БД» 42

ДПДД для состояния «Ввод данных» объекта «Операции над БД» 42

ДПДД для состояния «Сохранение БД» объекта «Операции над БД» 43

ДПДД для состояния «Неуспешное открытие БД» объекта «Операции над БД» 43

ДПДД для состояния «Обновление БД» объекта «Операции над БД» 44

Текст программы 47

Список использованной литературы 77

## Задание на курсовую работу

по дисциплине «Технология программирования»

тема:

Система рейтингого контроля знаний .

1.1 Задание

В данной курсовой работе было необходимо написать программу, с использованием объектно-ориентированного метода программирования и объектно-ориентированного анализа, которая позволяла бы вести учетную запись студентов по десяти предметам с оценкам. В программу заноситься Ф.И.О. студента, факультет на котором он учится, специальность, которую приобретает и группа соответственно. Программа автоматически считает средней балл и заносит его в базу. Данная программа сортирует, находит и фильтрует данные по студентам, что облегчает поиск или просмотр перед печатью нужных студентов. Результатом программы является вывод перед печатью всех студентов или по каждому студенту соответственно.

1.2 Реализация

Данная программа была реализована на объектно-ориентированном языке С++ и написана в среде разработчика программного обеспечения С++Builder.

###### Введение

Â îáúåêòíîå ïðîãðàììèðîâàíèå:

Òðàäèöèîííàÿ òåõíîëîãèÿ ïðîãðàììèðîâàíèÿ 70-õ ãîäîâ - ñòðóêòóðíîå ïðîãðàììèðîâàíèå:

 - ìîäóëüíîå ïðîãðàììèðîâàíèå;

 - íèñõîäÿùåå ïðîãðàììèðîâàíèå;

 - ñòðóêòóðíîå ïðîåêòèðîâàíèå ïðîöåäóð è äàííûõ (ïðîãðàììèðîâàíèå áåç goto).

 ßçûê Ïàñêàëü - ñîîòâåòñòâóåò óêàçàííûì ïðèíöèïàì è áûë ðàçðàáîòàí ïîä âëèÿíèåì èäåé ñòðóêòóðíîãî ïðîãðàììèðîâàíèÿ.

 Àëüòåðíàòèâíûé ïîäõîä - âîñõîäÿùåå ïðîãðàììèðîâàíèå - ïðåäïîëàãàåò â ïðîñòåéøåì ñëó÷àå ñîçäàíèå ñëîÿ ñòðóêòóð äàííûõ è ïðîöåäóð, îáåñïå÷èâàþùèõ ïîëíûé íàáîð äåéñòâèé íàä îáúåêòàìè, êîòîðûå ïðåäñòàâëåíû â äàííîé çàäà÷å. Ïðèìåð òðàäèöèîííîãî ïîäõîäà áèáëèîòåêà ñòàíäàðòíûõ ôóíêöèé.

 Ñëåäóþùèé øàã - ââåäåíèå â ïðîãðàììó îáúåêòîâ. Ïîä îáúåêòîì ïîíèìàåòñÿ ñòðóêòóðà äàííûõ, êîòîðàÿ ñîäåðæèò ïîëíóþ èíôîðìàöèþ î ñîñòîÿíèè ñîîòâåòñòâóþùåãî ôèçè÷åñêîãî îáúåêòà, êîòîðûé îòîáðàæàåòñÿ ïðîãðàììîé. Â Ñè ýòîìó ìîæåò ñîîòâåòñòâîâàòü ñòðóêòóðà struct, â Ïàñêàëå - çàïèñü (record). Ìíîæåñòâî îáúåêòîâ îäíîãî òèïà ñîñòàâëÿþò ïîíÿòèå êëàññà. Îáúåêòíî-îðèåíòèðîâàííûé ïîäõîä ê ðàçðàáîòêå ïðîãðàìì ïðåäïîëàãàåò, ÷òî â ïðîãðàììå óñòàíàâëèâàåòñÿ âçàèìíî-îäíîçíà÷íîå ñîîòâåòñòâèå ìåæäó ôèçè÷åñêèìè îáúåêòàìè,

îòîáðàæàåìûìè ïðîãðàììîé, è ïðîãðàìííûìè îáúåêòàìè, ÿâëÿþùèìèñÿ, ïî ñóùåñòâó, ñòðóêòóðèðîâàííûìè ïåðåìåííûìè (â äàëüíåéøåì ïîä òåðìèíîì "îáúåêò" áóäåì ïîíèìàòü ïðîãðàììíûé îáúåêò).

Òðàäèöèîííûé ïîäõîä: ---------- ïåðåìåííàÿ òèï äàííûõ

Îáúåêòíî-îðèåíòèðîâàííûé ïîäõîä:

ôèçè÷åñêèé ïðîãðàììíûé êëàññ îáúåêò îáúåêò îáúåêòîâ

 Ïðè ñîçäàíèè îáúåêòîâ ïðîãðàììèñò îïðåäåëÿåò ìíîæåñòâî ôóíêöèé, ïðè ïîìîùè êîòîðûõ (à òî÷íåå, èñêëþ÷èòåëüíî ÷åðåç êîòîðûå)íàä îáúåêòîì âûïîëíÿåòñÿ íåêîòîðîå äîïóñòèìîå ìíîæåñòâî îïåðàöèé.

Òàêèå ôóíêöèè äîëæíû èìåòü îáÿçàòåëüíûé ïàðàìåòð - ññûëêó íà òåêóùèé îáúåêò, äëÿ êîòîðîãî îíè âûçûâàþòñÿ. Ñàìè ôóíêöèè ÿâëÿþòñÿ íåîòúåìëèìîé ÷àñòüþ ïîíÿòèÿ êëàññà îáúåêòîâ, òàê êàê îíè îïðåäåëÿþò âîçìîæíûå äåéñòâèÿ íàä îáúåêòàìè îäíîãî è òîãî æå òèïà (òî åñòü êëàññà).

Ôàêòè÷åñêè îïðåäåëåíèå êëàññà îáúåêòîâ êàê òèïà äàííûõ è èçâåñòíîãî íàáîðà ôóíêöèé äëÿ âûïîëíåíèÿ îïåðàöèé íàä ïåðåìåííûìè ýòîãî òèïà ýêâèâàëåíòíî ïîíÿòèþ áàçîâîãî òèïà äàííûõ (ÁÒÄ) ÿçûêà ïðîãðàììèðîâàíèÿ. Åäèíñòâåííîå îòëè÷èå êëàññà îò ÁÒÄ çàêëþ÷àåòñÿ â òîì, ÷òî ïåðâûé îïðåäåëÿåòñÿ ïðîãðàììèñòîì, à âòîðîé âñòðîåí â îïðåäåëåíèå ÿçûêà ïðîãðàììèðîâàíèÿ.

 ßçûê ïðîãðàììèðîâàíèÿ Ñè++ ïðåäñòàâëÿåò ñîáîé ðàñøèðåíèå ÿçûêà Ñè äëÿ ïðîãðàììèðîâàíèÿ îáúåêòîâ è èõ êëàññîâ. Ïðè ýòîì èñïîëüçîâàíèå êëàññîâ ýêâèâàëåíòíî âïëîòü äî ñèíòàêñèñà èñïîëüçîâàíèþ áàçîâûõ òèïîâ äàííûõ.

# Глава I

# Системы поддержки принятия решений

**1. Система Paradox**

 Paradox представляет из себя реляционную базу данных, разработанную формой Borland. Данная база данных предназначена как для непрофессионалов, которым нужно решать задачи типа картотеки и пр., так как обладает достаточно развитой и понятной системой меню и подсказки, так и для профессиональных программистов, разрабатывающих сложные информационные системы, используя при этом язык запросов QBE или язык разработки приложений PAL. Кроме версий под DOS, разработана версия Paradox for Windows. Требования к техническим средствам и сравнительные характеристики по быстродействию см. Глава ??. Поддерживаются следующие типы полей:

 - алфавитно-цифровые (строковые);

 - числовые;

 - денежные;

 - дата;

 - Memo - поля;

 - графические;

 - OLE - поля;

 - BLOB (Binary Large OBject);

 Причем, типы полей от Memo до Blob поддерживаются только в Paradox for Windows.

 Строковые поля позволяют содержать любую последовательность символов длиной до 255 байт.

 Числовые поля могут содержать как целые, так и действительные числа с числов цифр до 15.

 Денежные поля - тоже что и числовые с двумя знаками после запятой, отдельный тип служит только для определения Paradox-ом формата вывода данных полей.

 Поля даты - соответственно содержат дату и позволяют проводить над собой соответствующие операции сравнения и вычитания.

 Мемо - поля содержат любой текст неограниченной длины (вместе с атрибутами текста - шрифт, начертание, цвет и др.)

 В графических полях могут содержаться графика типа bitmat.

 В OLE - полях могут содержаться объекты от других приложений Windows.

 В BLOB - полях можно хранить любые данные по выбору пользователя (на самом деле с точки зрения Paradox, Memo, Graphic и OLE - поля также являются полями типа BLOB). В настоящий момент возможно хранение объектов MultiMedia, чертежи и другие приложения.

 Достаточно удобным является тот факт, что для всех типов полей существует значение "пустое поле".

 Также следует отметить тип данных "BLOB", т.к. в полях такого типа можно хранить любые данные произвольного размера, что выгодно отличается от MEMO-полей в dBASE и Clipper, которые позволяют хранить только текст, да и то ограниченного об'ема (4K).

 Вся работа по созданию и работе с таблицами, подготовке отчетов и запросов, созданию достаточно сложных приложений, не требующих каких-либо сложных расчетов проводится в единой

интегрированной среде при помощи системы меню. Система обладает достаточно широкими возможностями по манипулированию данными и построению отчетов, некоторыми полезными статистическими и алгебраическими функциями для создания расчетных задач.

 Функции манипулирования данными можно разделить на следующие достаточно стандартные группы:

* работа с таблицами;
* работа с отдельными записями;

 - подготовки отчетов;

 - работа с запросами;

 - работа с сетью;

 - работа с макросами;

 - работы с деловой графикой.

 Функии работы с таблицами позволяют создавать таблицы, переименовывать, удалять, копировать и, что достаточно важно, модифицировать структуру таблиц, даже если в них уже находятся данные. Также, что немаловажно, все эти операции производятся не над отдельными файлами (как, скажем, в dBase), а над всем семейством таблиц (включая файлы данных, индексов, запросов и отчетов). Имеются и такие операции реляционной алгебры, как объединение и вычитание ?таблиц?отношений?.

 Функции работы с записями - стандартные (добавить, удалить, отредактировать).

 Удобными средствами обладает генератор отчетов - форму отчета можно "нарисовать" на экране и получить фактически то, что вы видите. Генератор отчетов позволяет создавать как отчеты в "стандартной" (табличной) форме, так и отчет в свободной форме (в т.ч. наклейки, деловые письма и пр.), производить группировку отчета по значениям полей (причем нескольких уровней вложенности), использовать вычисляемые поля.

 Функции запросов реализованы на реляционном языке QBE, причем отчеты можно формировать и по запросным формам. В Paradox for Windows построение запросов еще более

упрощено - для связей таблиц друг с другом достаточно просто провести линии между соответствующими полями.

 Функции работы с сетью позволяют получить доступ к удаленной БД, заблокировать данные для защиты от доступа других пользователей на время модификации, установить защиту по паролю от несанкционированного доступа.

 Функции работы с макросами позволяют практически создавать сценарий работы с пользователем и делать некоторые простые приложения.

 Система деловой графики позволяет по таблицам строить до шести функций в виде различного рода диаграмм и графиков.

 Если всех перечисленных возможностей недостаточно, то можно воспользоваться языком построения приложений PAL. Данный язык позволяет воспользоваться всеми перечисленными выше

возможностями, т.к. из него можно иметь доступ к функциям основного меню, а также обладает дополнительными средствами, характерными для языков программирования (управляющие структуры

типы if - then - else, циклы, переменные, операторы ввода-вывода, дополнительные математические и статистические функции, возможность создания процедур и библиотек процедур) и средствами доступа к продукту фирмы Borland - SQL-link. Средствами Pal можно создавать пользовательские меню, окна и

прочие радости для неискушенного пользователя, для которого картинка важнее, чем содержание.

 В системе имеется отладчик, с помощью которого можно достаточно удобно отлаживать программы на PAL. К сожалению, для работы программ на PAL необходима либо система Paradox, либо соответствующий модуль исполнения, т.е. проблематичной становится возможность распространения программ (каждый пользователь должен приобрести соответствующий Run-time модуль для исполнения программ.

 По показателям быстродействия система находится в первых рядах среди подобных.

 Система поддерживает следующие индексы - первичный (не может быть повторяющихся ключевых полей), вторичный - по одному полю, и составной - по группе полей. К сожалению, если таблица имеет несколько индексов, то при работе с записями производится модификация только одного индекса, а все остальные будут перестраиваться либо по окончании работы с таблицей, либо по запросу пользователя. Можно сделать этот процесс "прозрачным" для пользователя (первая возможность - индексы перестраиваются при закрытии таблицы), на ясно, что это достаточно сильно замедляет работу с подобными таблицами и причиняет пользователю некоторые неудобства (особенно если таблицы больших размеров).

 Система поддерживает доступ к БД на сервере по сети типа Novell, 3Com, Starlan, Banyan, а также другими, совместимыми с NetBios и обеспечивает необходимую блокировку данных как на уровне таблиц, так и на уровне отдельных записей. Paradox for Windows позволяет работать со всеми типами сетей, поддерживаемых Windows. Система имеет и такую полезную особенность, как возможность определить во время работы , менялись ли данные в таблицах, к которым имеют доступ несколько пользователей одновременно, хотя это и увеличивает нагрузку на сеть и может создать некоторые проблемы при наличии большого числа пользователей .

 Данные в таблицах хранятся в их естественном виде без преобразования, а не в виде символьных строк, как это принято в формате dBase, что сказывается на ускорении работы и уменьшении размеров самой базы данных.

 Имеются средства экспорта и импорта данных из/в формат Qattro/Pro, 1-2-3, Symphony, dBase, VisiCalc, а в системе Paradox for Windows возможно работать с файлами dBase без преобразования.

 Таким образом, система достаточна удобна как для неподготовленного пользователя, который хочет решить свою мелкую задачку типа записной книжки - за счет развитой системы меню и подсказки; и удобна для профессионального программиста, который решает достаточно сложные задачи - за счет богатых возможностей, предоставляемых системой, мощного реляционного языка запросов QBE, развитых средств построения отчетов, наличия языка построения приложений, имеющего выход на SQL-link; развитые

средства защиты данных от несанкционированного доступа. Все это позволяет строить сложные приложения, работающие в сети со многими пользователями. Но отсутствие возможности получить исполняемый модуль накладывает ограничения на широкое распространение разработанных программ (хотя Run-Time модуль стоит достаточно дешево, все-таки психологически лучше купить одну или несколько программ у одного изготовителя, чем связываться с несколькими) А работа фактически в режиме интерпретации PAL хоть и практически не сказывается при построении сложных информационныо-поисковых систем, в которых время на поиск данных значительно превышает время выполнения расчетов, но накладывает ограничения на системы, в которых кроме задач поиска стоят достаточно сложные задачи обработки данных. Но Paradox for Windows позволяет преодолеть это ограничение, т.к. в нем возможно иметь доступ к функциям, написанных на обычных языках программирования и собранные в динамические библиотеки (DDL). Так что можно пользоваться этой возможностью, хотя перенос программ в среду Windows (в том числе

и СУБД) работу этих программ явно не ускоряет.

 В пользу выбора данной системы для нашей задачи могут служить следующие факторы (наверное):

 - программирование на языке программирования PAL позволяет проводить достаточно сложную обработку данных, позволяет использовать развитые средства диалога с пользователем;

 - данные в БД хранятся в их естественном виде, что позволяет увеличить скорость обработки и уменьшить необходимое для хранения данных пространство;

 - очень полезные поля типа BLOB - позволяют хранить текст произвольной длины, например, тексты методичек и пр.;

 - возможность сетевого взаимодействия позволяет проводить дальнейшее расширение системы до уровня института с подключением других подсистем (скажем, УэО, деканат и пр.) и обеспечивать необходимую в рамках подобных систем защиту данных;

 - богатые возможности по экспорту и импорту данных;

 - возможность работы под Windows;

 - относительно простые, но мощные средства для построения запросов к базе данных.

 Хотя, конечно, вышеуказанные ограничения (невозможность получить .ЕХЕ - файл и интерпретатор) - это для нас явные минусы.

**Глава IV**

**Особенности объектно-ориентированного анализа**

При разработке данной курсовой работы был использован объектно-ориентированный подход к написанию данной программы. Проектирование происходило следующим образом:

1. Была сформирована информационная модель, на которой были выделены объекты, оказывающие влияние на ход процесса. В процессе проектирования намеренно в информационную модель были включены некоторые абстрактные типы данных, которые при дальнейшей разработке не образовали объекты. Это было сделано с целью более полно отобразить все отношения и взаимодействия, возникающие при работе программы.
2. Для каждого объекта была сформирована модель состояний, которая отражала события и состояния, в которых мог находиться объект, т.е. его жизненный цикл. Для нескольких объектов не было модели состояний. Это связано с тем, что характер поведения данных объектов является сугубо статическим и не претерпевает значительных изменений. Наоборот, объекты, имеющие динамическое поведение были полно отображены на моделях состояний. Особенность представленных моделей состояний заключается в том, что на них представлены события и действия, которые возникают при возникновении первых.
3. Модель взаимодействия объектов представляет собой графическое представление связей и взаимодействий между всеми объектами, определенными на информационной модели.
4. У объектов, с моделью состояний, была сформирована таблица переходов в состояния, демонстрирующая непротиворечивость модели состояний.
5. Для всех объектов была сформирована таблица всех возможных событий объектов.
6. Для каждого действия у каждого определенного объекта была сформирована ДПДД, отражающая последовательность действий, возникающих при определенном действии.

**Информационная модель**

Получить

данные

Содержит

\*ID Хранилища

\*ID Базы данных (R12)

\*Создание базы данных

1. Основной интерфейс (ОИ)

\*ID Операции

\*имя базы данных

\*путь к базе данных

\*статус

2. Операции над БД (ОБД)

\*ID псевдоним базы

\*число записей

\*статус

4. База данных (БД)

\*ID Базы данных (R6)

\*номер записи

\*описание действия

5. Запись в БД (ЗБД)

R9

R1

R12

R3

R8

R4

R6

Передача

количества

Запрос

Ответ

Обеспечивает

связь

Выполнение

запроса

Запрашивает

данные

Получить вектор

Содержится в

\*количество студентов

\*вектор-заполнитель

3. Печать (Х)

Отвечает на

запросы

Использует

данные

Возврат

результата

Запрос чистых

продукций

Выполнение

запроса

\*текущее число студентов

\*один студент

6. Просмотр пер. печ (Г)

**Модель состояний для объекта «Операции над БД»**

Порождает O6:

Редактирование

Порождает O1: Инициализация БД

5

Закрытие БД

6

Редактирование

БД

8

Ввод в Базу

9

Сохранение

7

Обновление

БД

O6: Осуществляется редактирование БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Редактирование)

O7: Осуществляется обновление БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Обновление)

O9: Осуществляется сохранение БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Сохранение)

O8: Осуществляется ввод строк

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Генерация)

O5: Осуществляется закрытие БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Закрытие)

Порождает O5: Закрытие БД

БД

2

Создание

базы

3

Открытие

базы

4

Неуспешное

открытие

 Создается объект на главной форме

 ID Основной Интерфейс, ID Операции

ID Основной Интерфейс.ID Операции.Путь к БД= ""

ID Основной Интерфейс.ID Операции.Имя БД= ""

Создается объект ввода данных

ID Основной Интерфейс.ID Операции.ID ввода данных

O3: Осуществляется открытие БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Открытие)

O2: Осуществляется создание БД

Вызывается операция

(ID Основной Интерфейс.ID

Операции.Создание)

Порождает O6: Редактирование БД

O3: Осуществляется открытие БД

Вызывается операция

(ID Основной Интерфейс.ID

 Операции.Открытие)

1

Инициализация

O1. Создается главная форма

(ID Основной Интерфейс)

**Операции**

O4: Неуспешное открытие

(ID Основной Интерфейс.ID

Операции.Открытие.Статус)

7

**Модель состояний для объекта «Ввод данных»**

2

Откл. Редактир.

Базы

3

Формирование

Поиска,фильтр.,сортир.

4

Формирование

групп

5

Отчет

6

Печать

Г3: Происходит просмотр введенных данных

 ID Основной Интерфейс.ID Операции над БД.ID Переход по строкам

Г4: Происходит формирование групп на основе введенных данных

 ID Основной Интерфейс.ID Операции над БД.ID Формировать по выбору

Г5: Происходит формирование готовых групп

 ID Основной Интерфейс.ID Операции над БД.ID .Фильтр по сортир., фил.

Г6: Объект печать

 ID Основной Интерфейс.ID Операции над БД.ID формирование групп

1

Инициализация

Г1. Создается объект ввода данных

(ID Основной Интерфейс.ID Операции над БД.ID ввода данных)

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Ф.И.О.

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Факультет

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Специальность

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Группа

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Предмет=10

ID Основной Интерфейс.ID Операции над БД.ID Ввод данных.Оценка=10

Г2: Из базы данных ID Основной Интерфейс.ID Операции

над БД.ID Запись в БД отключение редактирования

ID Основной Интерфейс.ID Операции над БД.сортировка, фильтрация, поиск

Ввод данных

**Модель взаимодействия объектов**

Гф2: Устоновить поле

Пользователь

Основной

интерфейс

Ввод данных

Просмотр перед печатью

Просморт данных

Запрос пользователя

Сообщение для пользователя

Д1:Построить

дерево

Гф1: Произвести ввод

Г2: Решение принято

Г1:Принять решение

O3:Получить имя БД

О9: Сохранить БД

О6:Редактировать БД

О2:Создать БД

О3:Открыть БД

О5:Закрыть БД

База данных

BDE

Аппаратные

абстракции

ОС

ЭВМ

АА(i):Запрос

AAB(i):Результат запроса

OS(i):Запрос

АА(i):Результат

запроса

БД (i):Запрос

OS(i):Результат запроса

BDE(i):Запрос

BDE(i):Результат запроса

Операции над БД

Перемещение по БД

Печать

Гф3: Ввести все данные

Х3:Сгенерировать количество

Г3.1:Получить по всем

Г3.2:Получить по одному

Г4.2: Получить длину группы

Г4.1: Получить размер группы

Г4.6:Сортировка

Г3.1:Фильтрация

Г3.2:Поиск

Х3.1: Вернуть опеч. лист

SYS1:Редактировать БД

SYS2:Создать БД

SYS3: Сохранить БД

SYS4:Открыть БД

SYS5:Закрыть БД

Ф4.1:Получить

Количеств студ.

# Таблица переходов в состояния

**для объекта «Операции над БД»**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Состояние | **O1** | **O2** | **O3** | **O4** | O5 | **O6** | **O7** | **O8** | **O9** |
| Инициализация | - | 2 | 3 | - | - | - | - | - | - |
| Создание | - | - | - | - | - | 6 | - | - | - |
| Открытие | - | - | - | 4 | - | 6 | - | - | - |
| Неуспех | - | - | - | - | 5 | - | - | - | - |
| Закрытие | 1 | - | - | - | - | - | - | - | - |
| Редактирование | - | - | - | - | - | - | 7 | - | - |
| Обновление | - | - | - | - | - | 6 | - | 8 | - |
| Ввод в базу | - | - | - | - | - | - | - | - | 9 |
| Схранение | 1 | - | - | - | - | - | - | - | - |

# Таблица переходов в состояния

**для объекта «Ввод данных»**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Состояние | **В1** | **В2** | **В3** | **В4** | **В5** | **В6** |
| Инициализация | - | 2 | - | - | - | - |
| Отключения редактирования | - | - | 3 | - | - | - |
| Формирование поиска, фильт., сортир., | - | - | - | 4 | - | - |
| Формирование групп | - | - | - | - | 5 | - |
| ПросмотрПеред печ. |  |  |  |  |  | 6 |
| Печать | - | - | - | - | - | - |

Список событий

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Метка | Значение | Данные события | Источник | Предназначение |
| О1 | Инициализация | ID Основной интерфейс.ID Операции | Основной интерфейс | Операции |
| О2 | Создание БД | ID Основной интерфейс.ID Операции.Создание | Операция | Создание |
| О3 | Открытие БД | ID Основной интерфейс.ID Операции.Открытие | Операция | Открытие |
| О4 | Неуспех | ID Основной интерфейс.ID Операции.Открытие.Статус | Операция | Статус |
| О5 | Закрытие БД | ID Основной интерфейс.ID Операции.Закрытие | Операция | Закрытие |
| О6 | Редактирование БД | ID Основной интерфейс.ID Операции.Редактирование | Операция | Редактирование |
| О7 | Обновление БД | ID Основной интерфейс.ID Операции.Обновление | Операция | Обновление |
| О8 | Ввод в БД | ID Основной интерфейс.ID Операции. | Операция | Редактирование |
| О9 | Сохранение БД | ID Основной интерфейс.ID Операции.Сохранение | Операция | Сохранение |
| В1 | Инициализация | ID Основной интерфейс.ID Операции  | Основной интерфейс | База данных |
| В2 | Отключения редактирования | ID Основной интерфейс.ID Операции. | Операция | Переход в основной интерфейс |
| В3 | Формирование поиска, фильтра, сортировки | ID Основной интерфейс.ID Операции.ID  | Основной интерфейс | Формирование фильтра |
| В4 | Формирование групп | ID Основной интерфейс.ID Операции. | Основной интерфейс | Формировать |
| В5 | Просмотр перед печатью | ID Основной интерфейс.ID Операции.ID  | Основной интерфейс | Просмотр учащихся |
| В6 | Печать | ID Основной интерфейс.ID Операции.ID  | Основной интерфейс | Учащихся |

**ДПДД для состояния Инициализация**

**объекта «Операции над БД»**



**ДПДД для состояния Создание базы**

**объекта «Операции над БД»**

О2.1 Сохранить

в новый файл

O2.2 Создать

таблицу

Атрибуты

файла

ID Таблицы

ID Файла

ID Основной

Интерфейс

ID Базы

О3.4: Вернуть

управление

ID Основной

Интерфейс

**ДПДД для состояния Открытие базы**

**объекта «Операции над БД»**



**ДПДД для состояния Редактирование базы**

**объекта «Операции над БД»**

ID Основной

Интерфейс

X2:

Ввод данных

X1: Инициал.

окно диалога

-

ID Базы

X1.2: Получить

данные

X1.3:Ошибка

заполнения

Ошибка

О3.4: Вернуть

управление

ID Основной

Интерфейс

ID Операции

**ДПДД для состояния Закрытие базы**

**объекта «Операции над БД»**



**ДПДД для состояния Сохранение БД**

**объекта «Операции над БД»**



**ДПДД для состояния Неуспешное открытие БД**

**объекта «Операции над БД»**



**ДПДД для состояния Обновление БД**

**объекта «Операции над БД»**



## Текст программы

В листинг программы был включен весь тот код, который был написан в процессе разработки и включения тех строк кода, которые были автоматически сгенерированы С++ Builder:

1. Заголовочные файлы:

**Kur1.h:**

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

#ifndef kur1H

#define kur1H

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\Menus.hpp>

#include <vcl\Dialogs.hpp>

#include <vcl\ExtCtrls.hpp>

#include <vcl\ComCtrls.hpp>

#include <vcl\DBGrids.hpp>

#include "Grids.hpp"

#include <vcl\Buttons.hpp>

#include <vcl\DBCtrls.hpp>

#include <vcl\Mask.hpp>

#include <vcl\DB.hpp>

#include <vcl\DBTables.hpp>

#include <vcl\QuickRep.hpp>

#include <vcl\system.hpp>

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

class TForm1 : public TForm

{

\_\_published: // IDE-managed Components

 TMainMenu \*MainMenu1;

 TMenuItem \*N1;

 TMenuItem \*N3;

 TMenuItem \*N4;

 TMenuItem \*N10;

 TMenuItem \*N14;

 TMenuItem \*N16;

 TMenuItem \*N17;

 TOpenDialog \*OpenDialog1;

 TSaveDialog \*SaveDialog1;

 TMenuItem \*N18;

 TMenuItem \*N20;

 TPrintDialog \*PrintDialog1;

 TPrinterSetupDialog \*PrinterSetupDialog1;

 TStatusBar \*StatusBar1;

 TPanel \*Panel1;

 TDBGrid \*DBGrid1;

 TDBNavigator \*DBNavigator1;

 TMenuItem \*N2;

 TMenuItem \*N5;

 TMenuItem \*N6;

 TBitBtn \*BitBtn1;

 TBitBtn \*BitBtn2;

 TBitBtn \*BitBtn3;

 TMenuItem \*N7;

 void \_\_fastcall N10Click(TObject \*Sender);

 void \_\_fastcall N17Click(TObject \*Sender);

 void \_\_fastcall N14Click(TObject \*Sender);

 void \_\_fastcall N5Click(TObject \*Sender);

 void \_\_fastcall N2Click(TObject \*Sender);

 void \_\_fastcall N16Click(TObject \*Sender);

 void \_\_fastcall N20Click(TObject \*Sender);

 void \_\_fastcall BitBtn1Click(TObject \*Sender);

 void \_\_fastcall BitBtn2Click(TObject \*Sender);

 void \_\_fastcall BitBtn3Click(TObject \*Sender);

 void \_\_fastcall N7Click(TObject \*Sender);

 void \_\_fastcall N4Click(TObject \*Sender);

private: // User declarations

public: // User declarations

// AnsiString file2;

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

};

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

extern TForm1 \*Form1;

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

#endif

**Datam.h:**

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

#ifndef DatamH

#define DatamH

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\DB.hpp>

#include <vcl\DBTables.hpp>

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

class TDM1 : public TDataModule

{

\_\_published: // IDE-managed Components

 TDataSource \*DataSource1;

 TTable \*Table1;

 TQuery \*Query1;

 TDataSource \*DataSource2;

private: // User declarations

public: // User declarations

 \_\_fastcall TDM1(TComponent\* Owner);

};

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

extern TDM1 \*DM1;

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

#endif

**filename.h:**

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

#ifndef filenameH

#define filenameH

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\ExtCtrls.hpp>

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

class TForm6 : public TForm

{

\_\_published: // IDE-managed Components

 TEdit \*Edit1;

 TLabel \*Label2;

 TBevel \*Bevel1;

 TButton \*Button1;

 TButton \*Button2;

 void \_\_fastcall Button1Click(TObject \*Sender);

private: // User declarations

public: // User declarations

 \_\_fastcall TForm6(TComponent\* Owner);

};

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

extern TForm6 \*Form6;

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

#endif

**filter.h:**

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

#ifndef filterH

#define filterH

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\ExtCtrls.hpp>

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

class TForm5 : public TForm

{

\_\_published: // IDE-managed Components

 TBevel \*Bevel1;

 TLabel \*Label1;

 TLabel \*Label2;

 TComboBox \*ComboBox1;

 TEdit \*Edit1;

 TButton \*Button1;

 TButton \*Button2;

 TButton \*Button3;

 void \_\_fastcall Button3Click(TObject \*Sender);

private: // User declarations

public: // User declarations

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

};

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

extern TForm5 \*Form5;

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

#endif

**Findcust.h:**

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

#ifndef filterH

#define filterH

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\ExtCtrls.hpp>

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

class TForm5 : public TForm

{

\_\_published: // IDE-managed Components

 TBevel \*Bevel1;

 TLabel \*Label1;

 TLabel \*Label2;

 TComboBox \*ComboBox1;

 TEdit \*Edit1;

 TButton \*Button1;

 TButton \*Button2;

 TButton \*Button3;

 void \_\_fastcall Button3Click(TObject \*Sender);

private: // User declarations

public: // User declarations

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

};

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

extern TForm5 \*Form5;

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

#endif

**Repopt.h:**

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

#ifndef ReportH

#define ReportH

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

#include <vcl\ExtCtrls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Quickrep.hpp>

#include <vcl\Dialogs.hpp>

#include <vcl\Forms.hpp>

#include <vcl\Controls.hpp>

#include <vcl\Graphics.hpp>

#include <vcl\Classes.hpp>

#include <vcl\SysUtils.hpp>

#include <vcl\Messages.hpp>

#include <vcl\Windows.hpp>

#include <vcl\System.hpp>

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

class TQRListForm : public TForm

{

\_\_published:

 TQuickReport \*QuickReport;

 TQRBand \*Title;

 TQRBand \*PageHeader;

 TQRBand \*Detail;

 TQRBand \*PageFooter;

 TQRLabel \*QRLabel4;

 TQRLabel \*QRLabel1;

 TQRLabel \*QRLabel2;

 TQRLabel \*QRLabel3;

 TQRLabel \*QRLabel5;

 TQRLabel \*QRLabel6;

 TQRGroup \*QRGroup1;

 TQRSysData \*QRSysData1;

 TQRShape \*QRShape1;

 TQRDBText \*QRDBText3;

 TQRDBText \*QRDBText1;

 TQRDBText \*QRDBText9;

 TQRDBText \*QRDBText10;

 TQRDBText \*QRDBText2;

 void \_\_fastcall QRGroup1NeedData(AnsiString &Value);

private:

public:

 virtual \_\_fastcall TQRListForm(TComponent\* AOwner);

};

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

extern TQRListForm \*QRListForm;

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

#endif

**RichАbt.h:**

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

#ifndef RichAbtH

#define RichAbtH

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

#include <ExtCtrls.hpp>

#include <StdCtrls.hpp>

#include <Dialogs.hpp>

#include <Forms.hpp>

#include <Controls.hpp>

#include <Graphics.hpp>

#include <Classes.hpp>

#include <SysUtils.hpp>

#include <Messages.hpp>

#include <Windows.hpp>

#include <System.hpp>

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

class TForm2 : public TForm

{

\_\_published:

 TImage \*ProgramIcon;

 TButton \*OKButton;

public:

 virtual \_\_fastcall TForm2(TComponent \*Owner);

};

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

extern TForm2 \*Form2;

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

#endif

**Sort.h:**

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

#ifndef SortH

#define SortH

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\ExtCtrls.hpp>

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

class TForm4 : public TForm

{

\_\_published: // IDE-managed Components

 TBevel \*Bevel1;

 TComboBox \*ComboBox1;

 TLabel \*Label1;

 TButton \*Button1;

 TButton \*Button2;

 void \_\_fastcall ComboBox1Change(TObject \*Sender);

private: // User declarations

public: // User declarations

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

};

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

extern TForm4 \*Form4;

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

#endif

**Unit1.h:**

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

#ifndef Unit1H

#define Unit1H

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

#include <vcl\Classes.hpp>

#include <vcl\Controls.hpp>

#include <vcl\StdCtrls.hpp>

#include <vcl\Forms.hpp>

#include <vcl\DBTables.hpp>

#include <vcl\DB.hpp>

#include <vcl\DBCtrls.hpp>

#include <vcl\Mask.hpp>

#include <vcl\Buttons.hpp>

#include <vcl\ComCtrls.hpp>

#include <vcl\DBGrids.hpp>

#include "Grids.hpp"

#include <vcl\ExtCtrls.hpp>

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

class TForm3 : public TForm

{

\_\_published: // IDE-managed Components

 TLabel \*Label1;

 TLabel \*Label2;

 TLabel \*Label3;

 TLabel \*Label4;

 TDBEdit \*DBEdit3;

 TDBEdit \*DBEdit6;

 TLabel \*Label6;

 TLabel \*Label7;

 TDBEdit \*DBEdit8;

 TButton \*Button1;

 TButton \*Button2;

 TButton \*Button5;

 TLabel \*Label5;

 TDBEdit \*DBEdit4;

 TDBEdit \*DBEdit1;

 TDBEdit \*DBEdit2;

 TLabel \*Label8;

 TDBEdit \*DBEdit5;

 TBitBtn \*BitBtn1;

 TBitBtn \*BitBtn2;

 TButton \*Button3;

 TStatusBar \*StatusBar1;

 void \_\_fastcall FormCreate(TObject \*Sender);

 void \_\_fastcall Button1Click(TObject \*Sender);

 void \_\_fastcall Button2Click(TObject \*Sender);

 void \_\_fastcall Button6Click(TObject \*Sender);

 void \_\_fastcall Button4Click(TObject \*Sender);

 void \_\_fastcall Button5Click(TObject \*Sender);

 void \_\_fastcall Table1AfterOpen(TDataSet \*DataSet);

 void \_\_fastcall Button3Click(TObject \*Sender);

private: // User declarations

public: // User declarations

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

};

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

extern TForm3 \*Form3;

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

#endif

1. Файлы с методами:

**Project1.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

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

USEFORM("kur1.cpp", Form1);

USEFORM("Richabt.cpp", Form2);

USERES("Project1.res");

USEFORM("Report.cpp", QRListForm);

USEDATAMODULE("Datam.cpp", DM1);

USEFORM("Unit1.cpp", Form3);

USEFORM("Findcust.cpp", fmFindCust);

USEFORM("Sort.cpp", Form4);

USEFORM("filter.cpp", Form5);

USEFORM("filename.cpp", Form6);

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

WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

 try

 {

 Application->Initialize();

 Application->CreateForm(\_\_classid(TForm1), &Form1);

 Application->CreateForm(\_\_classid(TQRListForm), &QRListForm);

 Application->CreateForm(\_\_classid(TDM1), &DM1);

 Application->CreateForm(\_\_classid(TForm3), &Form3);

 Application->CreateForm(\_\_classid(TfmFindCust), &fmFindCust);

 Application->CreateForm(\_\_classid(TForm4), &Form4);

 Application->CreateForm(\_\_classid(TForm5), &Form5);

 Application->CreateForm(\_\_classid(TForm6), &Form6);

 Application->Run();

 }

 catch (Exception &exception)

 {

 Application->ShowException(&exception);

 }

 return 0;

}

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

**Kur1.cpp:**

#include <vcl\vcl.h>

#include <stdio.h>

#pragma hdrstop

#include "kur1.h"

#include "RichAbt.h"

#include "Datam.h"

#include "Report.h"

#include "Unit1.h"

#include "Findcust.h"

#include "Sort.h"

#include "filter.h"

#include "filename.h"

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

#pragma link "Grids"

#pragma resource "\*.dfm"

TForm1 \*Form1;

AnsiString ff;

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

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

 : TForm(Owner)

{

 ff=Form1->Caption;

}

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

void \_\_fastcall TForm1::N10Click(TObject \*Sender)

{

 Form1->Caption=ff;

 Form6->ShowModal();

 N2->Enabled=true;

 N20->Enabled=true;

 N5->Enabled=true;

 N14->Enabled=true;

 N16->Enabled=true;

 Form1->Caption =Form1->Caption+"->"+DM1->Table1->TableName;

}

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

void \_\_fastcall TForm1::N17Click(TObject \*Sender)

{

Close();

}

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

void \_\_fastcall TForm1::N14Click(TObject \*Sender)

{

 QRListForm->QuickReport->Print();

}

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

void \_\_fastcall TForm1::N5Click(TObject \*Sender)

{

 DBGrid1->Visible=false;

 DBNavigator1->Enabled=false;

 BitBtn1->Enabled=false;

 BitBtn2->Enabled=false;

 BitBtn3->Enabled=false;

}

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

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

{

 DBGrid1->Visible=true;

 DBNavigator1->Enabled=true;

 BitBtn1->Enabled=true;

 DBGrid1->ReadOnly=true;

 BitBtn2->Enabled=true;

 BitBtn3->Enabled=true;

}

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

void \_\_fastcall TForm1::N16Click(TObject \*Sender)

{

 QRListForm->QuickReport->Preview();

}

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

void \_\_fastcall TForm1::N20Click(TObject \*Sender)

{

 Form3->ShowModal();

 DM1->Table1->Refresh();

 }

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

void \_\_fastcall TForm1::BitBtn1Click(TObject \*Sender)

{

 DBGrid1->DataSource=DM1->DataSource2;

 fmFindCust->ShowModal();

 if (fmFindCust->ModalResult == mrOk)

 DM1->Table1->GotoCurrent(DM1->Table1);

}

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

void \_\_fastcall TForm1::BitBtn2Click(TObject \*Sender)

{

 DBGrid1->DataSource=DM1->DataSource1;

 Form4->ShowModal();

 if (Form4->ModalResult == mrOk)

 DM1->Table1->GotoCurrent(DM1->Table1);

}

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

void \_\_fastcall TForm1::BitBtn3Click(TObject \*Sender)

{

 DBGrid1->DataSource=DM1->DataSource1;

 Form5->ShowModal();

 if (Form4->ModalResult == mrOk)

 DM1->Table1->GotoCurrent(DM1->Table1);

}

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

void \_\_fastcall TForm1::N7Click(TObject \*Sender)

{

 if ( OpenDialog1->Execute() )

 {

 Form1->Caption=ff;

 N2->Enabled=true;

 N20->Enabled=true;

 N5->Enabled=true;

 N14->Enabled=true;

 N16->Enabled=true;

 if (DM1->Table1->Active == true )

 DM1->Table1->Active = false;

 DM1->Table1->DatabaseName = ExtractFilePath( OpenDialog1->FileName );

 DM1->Table1->TableName = ExtractFileName( OpenDialog1->FileName );

 DM1->Table1->Active = true;

 Form1->Caption =Form1->Caption+"->"+DM1->Table1->TableName;

 }

}

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

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

{

Form2= new TForm2(Application);

Form2->ShowModal();

delete Form2;

}

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

**Datam.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "Datam.h"

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

#pragma resource "\*.dfm"

TDM1 \*DM1;

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

\_\_fastcall TDM1::TDM1(TComponent\* Owner)

 : TDataModule(Owner)

{

}

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

**filename.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "filename.h"

#include "Datam.h"

#include "kur1.h"

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

#pragma resource "\*.dfm"

TForm6 \*Form6;

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

\_\_fastcall TForm6::TForm6(TComponent\* Owner)

 : TForm(Owner)

{

}

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

void \_\_fastcall TForm6::Button1Click(TObject \*Sender)

{

DM1->Table1->Active=false;

DM1->Query1->Active=false;

if (Edit1->Text == "")

Edit1->Text="ITF.db";

TFileStream \*Old = new TFileStream("baza1.db", fmOpenRead);

TFileStream \*New = new TFileStream(Edit1->Text,fmCreate);

New->CopyFrom(Old, Old->Size);

delete Old; delete New;

DM1->Table1->TableName=Edit1->Text;

DM1->Query1->Active=true;

DM1->Table1->Active=true;

}

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

**filter.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "filter.h"

#include "Datam.h"

#include "kur1.h"

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

#pragma resource "\*.dfm"

TForm5 \*Form5;

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

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

 : TForm(Owner)

{

}

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

void \_\_fastcall TForm5::Button3Click(TObject \*Sender)

{

AnsiString filter;

filter=Edit1->Text;

if(ComboBox1->Text != ""){

if(Edit1->Text != "" ){

 if (ComboBox1->Text == "L.L.+."){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Where A=\""+filter+"\"");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "Lръєы№ЄхЄ"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Where W=\""+filter+"\"");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "TяхЎшры№эюёЄ№"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Where Q=\""+filter+"\"");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "+Ёєяяр"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Where S=\""+filter+"\"");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "TЁ.+рыы"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Where Z>="+filter);

 DM1->Query1->Open();

 DM1->Table1->Open();}}

 else

 {

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza ");

 DM1->Query1->Open();

 DM1->Table1->Open();}}

 else{

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza ");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 DM1->Table1->Refresh();}

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

**findcust.cpp:**

#include <vcl.h>

#pragma hdrstop

#include "FindCust.h"

#include "Datam.h"

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

#pragma resource "\*.dfm"

TfmFindCust\* fmFindCust;

char field[2]={'A','Z'};

char w1;

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

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

 : TForm(Owner)

{

}

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

void \_\_fastcall TfmFindCust::ComboBox1Change(TObject \*Sender)

{

 if (ComboBox1->Text == "L.L.+.")

 w1=field[0];

 else if (ComboBox1->Text == "TЁ.+рыы")

 {w1=field[1];

 //Edit1->Text = "";

 }

 else

 ComboBox1->Text ="L.L.+.";

 w1=field[0];

}

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

void \_\_fastcall TfmFindCust::Edit1Change(TObject \*Sender)

{

 bool Success;

 Set<TLocateOption,0,1> flags;

 flags << loCaseInsensitive << loPartialKey;

 TLocateOptions Options;

 Success = DM1->Table1->Locate(w1,Edit1->Text,flags);

 DM1->Table1->Refresh();

}

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

**Report.cpp:**

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

#include <vcl.h>

#pragma hdrstop

#include "Report.h"

#include "Datam.h"

#include "kur1.h"

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

#pragma resource "\*.dfm"

TQRListForm \*QRListForm;

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

// QuickReport simple list

// - Connect a datasource to the QuickReport component

// - Put QRDBText components on the Detail band

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

\_\_fastcall TQRListForm::TQRListForm(TComponent\* AOwner)

 : TForm(AOwner)

{

}

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

void \_\_fastcall TQRListForm::QRGroup1NeedData(AnsiString &Value)

{

//QuickReport->NewPage();

}

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

**RichAbt.cpp:**

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

//Borland C++Builder

//Copyright (c) 1987, 1997 Borland International Inc. All Rights Reserved.

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

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

#include <vcl.h>

#pragma hdrstop

#include "RichAbt.h"

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

#pragma resource "\*.dfm"

TForm2 \*Form2;

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

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

 : TForm(Owner)

{

}

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

**Sort.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "Sort.h"

#include "Datam.h"

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

#pragma resource "\*.dfm"

TForm4 \*Form4;

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

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

 : TForm(Owner)

{

}

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

void \_\_fastcall TForm4::ComboBox1Change(TObject \*Sender)

{

 if (ComboBox1->Text == "L.L.+."){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Order by A asc");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "Lръєы№ЄхЄ"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Order by W asc");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "TяхЎшры№эюёЄ№"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Order by Q asc");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "+Ёєяяр"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Order by S asc");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 else if (ComboBox1->Text == "Lръєы№ЄхЄ"){

 DM1->Query1->SQL->Clear();

 DM1->Query1->SQL->Append("Select \* From baza Order by Z asc");

 DM1->Query1->Open();

 DM1->Table1->Open();}

 DM1->Table1->Refresh();

 }

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

**Unit1.cpp:**

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

#include <vcl\vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "Datam.h"

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

#pragma link "Grids"

#pragma resource "\*.dfm"

TForm3 \*Form3;

char predmet1[9]={'H','N','U','J','M','I','K','1','O'};

char predmet2[9]={'E','D','C','R','F','V','T','G','B'};

double ozenka1[12];

int i=0;

double r;

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

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

 : TForm(Owner)

{

}

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

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

{

//-юсрты хь эютє¦ чряшё№ т срчє

DM1->Table1->Append();

}

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

void \_\_fastcall TForm3::Button1Click(TObject \*Sender)

{

 if(DM1->Table1->State == dsInsert||DM1->Table1->State == dsEdit)

 DM1->Table1->Post();

 DM1->Table1->Append();

 for(i=0;i<=9;i++){

 predmet1[i]=0;}

 i=0;

 Button5->Enabled=true;

 Button2->Enabled=true;

 DBEdit8->DataField='x';

 DBEdit5->DataField='y';

 DBEdit5->ReadOnly=false;

 DBEdit8->ReadOnly=false;

 DBEdit1->ReadOnly=false;

 DBEdit2->ReadOnly=false;

 DBEdit4->ReadOnly=false;

 DBEdit6->ReadOnly=false;

 DBEdit4->Visible=false;

 Label8->Visible=false;

 DM1->Table1->Refresh();

}

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

void \_\_fastcall TForm3::Button6Click(TObject \*Sender)

{

 if ( !DM1->Table1->Bof )

 DM1->Table1->Prior();

 i=10;

}

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

void \_\_fastcall TForm3::Button4Click(TObject \*Sender)

{

 if ( !DM1->Table1->Eof )

 DM1->Table1->Next();

 i=10;

}

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

void \_\_fastcall TForm3::Button2Click(TObject \*Sender)

{

 DM1->Table1->Delete();

 DM1->Table1->Refresh();

 if(DM1->Table1->Eof && DM1->Table1->Bof )

 Button2->Enabled=false;

 DBEdit4->Visible=false;

 Label8->Visible=false;

}

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

void \_\_fastcall TForm3::Button5Click(TObject \*Sender)

{

 if(i<=8)

{

 int Size = DBEdit8->GetTextLen();

 char \*name = new char[++Size];

 DBEdit8->GetTextBuf(name,Size);

 DBEdit8->Text = name;

 r=atoi(name);

 ozenka1[i]=r;

 delete name;

 r=0;

 DBEdit5->DataField=predmet1[i];

 DBEdit8->DataField=predmet2[i];

 i++;

 Button3->Enabled=true;

}

else {if(i==9){

 int Size = DBEdit8->GetTextLen();

 char \*name = new char[++Size];

 DBEdit8->GetTextBuf(name,Size);

 DBEdit8->Text = name;

 r=atoi(name);

 ozenka1[i]=r;

 delete name;

 r=0;}

ozenka1[10]=(ozenka1[0]+ozenka1[1]+ozenka1[2]+ozenka1[3]+ozenka1[4]+ozenka1[5]+ozenka1[6]+ozenka1[7]+ozenka1[8]+ozenka1[9])/10;

DBEdit4->SelText=ozenka1[10];

DM1->Table1->FieldValues["Z"] = ozenka1[10];

Button5->Enabled=false;

Button3->Enabled=false;

DBEdit5->ReadOnly=true;

DBEdit8->ReadOnly=true;

DBEdit1->ReadOnly=true;

DBEdit2->ReadOnly=true;

DBEdit4->ReadOnly=true;

DBEdit6->ReadOnly=true;

DBEdit4->Visible=true;

Label8->Visible=true;

}}

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

void \_\_fastcall TForm3::Table1AfterOpen(TDataSet \*DataSet)

{

 if(DM1->Table1->Eof && DM1->Table1->Bof )

 Button2->Enabled=false;

}

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

void \_\_fastcall TForm3::Button3Click(TObject \*Sender)

{

 if(i>=0)

{i--;

DBEdit5->DataField=predmet1[i];

DBEdit8->DataField=predmet2[i];

}

else

Button3->Enabled=false;

Button5->Enabled=true;}

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

## Список использованной литературы

1. Трахтенгерц Э.А. «Компьютерная поддержка принятия решений»
2. Том Сван «Программирование в Borland C++ 5.02»
3. «Объектно-ориентированный анализ»
4. Мэт Теллес «Borland С++ Builder»