МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

ГЛАВНОЕ УПРАВЛЕНИЕ ПО ОБРАЗОВАНИЮ

МОГИЛЕВСКОГО ОБЛАСТНОГО ИСПОЛНИТЕЛЬНОГО КОМИТЕТА

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«МОГИЛЕВСКИЙ ГОСУДАРСТВЕННЫЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»

Специальность 2-40 01 01

Учебная группа ПО-455

Учебная дисциплина Основы алгоритмизации

и программирования

**КУРСОВОЙ ПРОЕКТ**

**РАЗРАБОТКА** **ПРОГРАММНОГО МОДУЛЯ «АЭРОПОРТ»**

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КП ПО.00.00.000 ПЗ**

Разработал A.О.Савич

Руководитель проекта Ю.А.Кашпарова

2021

Содержание

Введение 4

1. Общая характеристика задачи 5

2. Проектирование программного модуля 6

2.1 Описание входных и выходных данных 6

2.2 Алгоритм решения задачи 6

3. Реализация программного модуля 8

3.1 Описание структуры разрабатываемой задачи 8

3.2 Проектирование и реализация интерфейса задачи 8

3.3 Разработка справочной системы 9

3.4 Тестирование задачи 9

4. Мероприятия по энерго- и ресурсосбережению 10

Заключение 12

Список используемых источников 13

Приложение А (обязательное) Текст программы 14

### **Введение**

В последнее время резко возрос интерес к программированию. Это связано с развитием и внедрением в повседневную жизнь информационно-коммуникационных технологий. Если человек имеет дело с компьютером, то рано или поздно у него возникает желание, а иногда и необходимость, программировать. Среди пользователей персональных компьютеров в настоящее время наиболее популярно семейство операционных систем Windows и, естественно, что тот, кто собирается программировать, стремится писать программы, которые будут работать в этих системах. Несколько лет назад рядовому программисту оставалось только мечтать о создании собственных программ, работающих в среде Windows, т.к. единственным средством разработки был Borland C++ for Windows, явно ориентированный на профессионалов, обладающих серьезными знаниями и опытом.

Бурное развитие вычислительной техники, потребность в эффективных средствах разработки программного обеспечения привели к появлению систем программирования, ориентированных на так называемую «быструю разработку», среди которых можно выделить Borland Delphi и Microsoft Visual Basic. В основе систем быстрой разработки (RAD-систем, Rapid Application Development - среда быстрой разработки приложений) лежит технология визуального проектирования и событийного программирования, суть которой заключается в том, что среда разработки берет на себя большую часть рутинной работы, оставляя программисту работу по конструированию диалоговых окон и функций обработки событий. Производительность программиста при использовании RAD-систем - очень высокая.

Delphi - это среда быстрой разработки, в которой в качестве языка программирования используется язык Delphi. Язык Delphi - строго типизированный объектно-ориентированный язык, в основе которого лежит хорошо знакомый программистам Object Pascal.

### **1 Общая характеристика задачи**

Эффективная работа каждого предприятия зависит от оперативности и рациональности выполняемых действий. В настоящее время для хозяйственного субъекта очень важно быть конкурентно способным, особенно в сфере предоставления услуг. От качества предоставляемых услуг напрямую зависит имидж предприятия среди клиентов, а также авторитет среди конкурирующих организаций на рынке.

Одной из самых значимых сфер оказания услуг является сфера предоставления транспортных услуг. Ежедневно в мире перевозятся миллионы тонн грузов и сотни тысяч пассажиров. Наиболее быстрыми, хотя и не дешёвыми, являются авиаперевозки. Клиентам предоставляется возможность за очень короткое время преодолеть сравнительно большие расстояния.

При работе с пассажирами важно оперативно и точно предоставить информацию об интересующих их направлениях. Клиент может узнать о существующих рейсах, времени их вылета и прибытия, количества свободных мест в разных салонах самолёта.

В результате выполнения курсового проекта требуется разработать программу, позволяющую оперативно вести продажу авиабилетов в день вылета и их возврат, а также закрепить на практике знания предмета «Основы алгоритмизации и программирования», научится самостоятельно разрабатывать и тестировать приложения.

Программа должна позволять отображать список рейсов из базы данных: количество мест в первом и втором классе, также время вылета и текущую дату. Также в Программе должна присутствовать возможность вноса новых рейсов в базу данных и статистика проданных билетов.

Пользователь программы (кассир аэропорта) должен получить от клиента ряд сведений, необходимых для оформления заказа. А именно: дату вылета самолёта, номер рейса (хранящий в себе пункт назначения) и класс желаемого места. Билет можно купить только на текущую дату. А для сдачи билета в кассу необходимо открыть меню возврата и выбрать на основании сведений о дате вылета и номере рейса билет и удалить сведения о нем. После проведения операции продажи или возврата билета информация о билете отражается в базе.

В своей основе программа имеет базу данных Microsoft Access «airport.db».

Список функций:

- выполнение операции продажи авиабилета;

- выполнение операции возврата авиабилета;

- удаление из базы авиарейса;

- занесение в базу авиарейса;

- вывод статистики продаж.

Значительным преимуществом автоматизации данных операций является скорость их выполнения, возможность в любой момент осуществить запрос к данным, находящимся в базе и получить по этому запросу исчерпывающую информацию. Программа позволяет отказаться от необходимости ведения бумажных архивов.

### **2. Проектирование программного модуля**

### **2.1 Описание входных и выходных данных**

Впрограмму из таблицы «Cегодня» базы данных airport.db входят данные представленные в таблице 1.

Таблица 1 – Данные таблицы «Сегодня»

|  |  |
| --- | --- |
| Имя поля таблицы | Тип поля |
| Datin | Дата/Время |
| Board\_Number | Текстовый |
| Depart\_Time | Дата/Время |
| First\_Class | Числовой |
| Second\_Class | Числовой |

Из таблицы «Продажи» в программу входят данные представленные в таблице 2.

Таблица 2 – Данные таблицы «Продажи»

|  |  |
| --- | --- |
| Имя поля таблицы | Тип поля |
| Sale\_Date | Дата/Время |
| Board\_Number | Текстовый |
| Сlass | Текстовый |

Данные, которые отправляются в базу данных из программы представлены в таблице 3.

Таблица 3 – Данные, которые отправляются в базу данных из программы

|  |  |  |  |
| --- | --- | --- | --- |
| Выход | Имя Таблицы | Имя Поля Таблицы | Тип Поля |
| Выход в | Сегодня | Datin | Дата/Время |
| Выход в | Сегодня | Board\_Number | Текстовый |
| Выход в | Сегодня | Depart\_Time | Дата/Время |
| Выход в | Сегодня | First\_Class | Числовой |
| Выход в | Сегодня | Second\_Class | Числовой |
| Выход в | Продажи | Sale\_Date | Дата/Время |
| Выход в | Продажи | Board\_Number | Текстовый |
| Выход в | Продажи | Сlass | Текстовый |

### **2.2 Алгоритм решения задачи**

Общий алгоритм решения поставленной задачи включает в себя ряд пунктов:

1. Описание и создание структуры базы данных;

2. Разработка пользовательского приложения:

а) создание подключения к базе данных средствами Delphi;

б) создание экранных форм и помещение на них элементов для отображения и управления данными базы.

Структура базы данных «Аэропорт» была описана выше.

При разработке пользовательского приложения я использовал объектно-ориентированный язык программирования Delphi.

Согласно алгоритму реализации поставленной задачи вначале необходимо выполнить подключение к базе данных. Для этого мы создаём модуль и помещаем в него объект AdoConnection, служащий для подключения к какой-либо базе данных. Далее задаём свойства этого объекта. В свойстве AdoConnection дважды кликаем на connectionstring затем в окне данного свойства (рисунок 1).

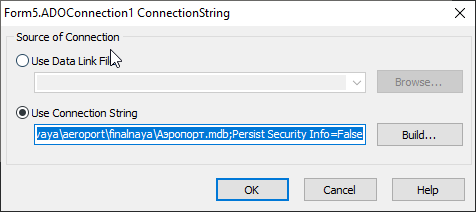


Рисунок 1 – Окно свойства ConnectionString

Нажимаем кнопку build и в этом окне необходимо выбрать поставщика OLE DB, непосредственно саму базу и права доступа к этой базе.

Окно выбора поставщика данных представлено на рисунке 2.

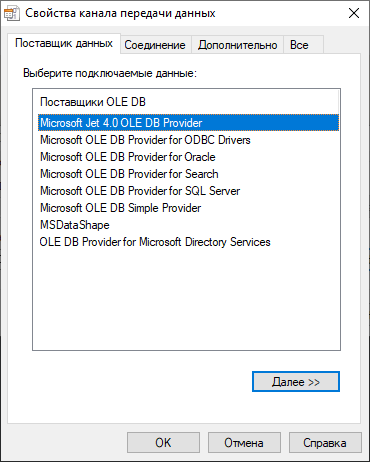


Рисунок 2 – Окно выбора поставщика данных

Окно соединения с базой данных представлено на рисунке 3.

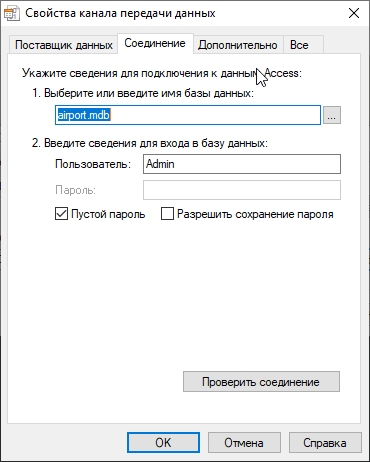


Рисунок 3 – Окно соединения с базой данных

После выбора базы необходимо проверить подключение нажатием кнопки «Проверить подключение» на вкладке «Подключение».

В свойстве LoginPrompt отключаем запрос пароля на доступ к базе, задав значение false. Для чтения данных из базы в модуль проекта помещаем компонент ADOTable.

С вкладки DataAccess в модуль InfoReisy.pas помещаем один обьект Datasource и для него в свойстве DataSet выбираем соответственно объект ADOTable.

На этом подключение к базе данных завершено.

Далее создаём ряд экранных форм. Для этого в Delphi существует команда New Form. Как и любой другой объект обладает набором свойств: Name (программное имя), Caption (строка заголовка), Height (высота), Width (ширина), Border Style (тип границы) и другие.  
На каждой форме, по необходимости, будем помещать компоненты нужные для реализации функционала данной формы. Общая схема программы представлена на рисунке 4.

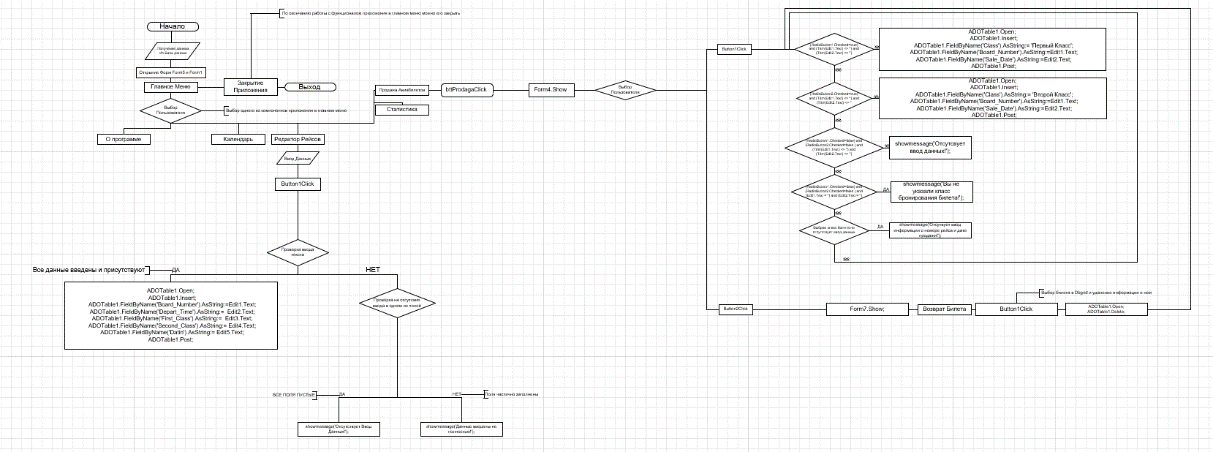


Рисунок 4 – Общая схема программы

### **3. Реализация программного модуля.**

### **3.1 Описание структуры разрабатываемой задачи.**

Весь курсовой проект выполнен в Delphi. Вся программа храниться в папке «Курсовой проект». В этой папке хранятся все Unit и Project, а также другие файлы, необходимые для корректной работы программы.

В нашей программе форм будет 8, служащих для реализации функционала, описанного в проектировании программного модуля:

- Form5 – является главной Формой программы и используется для показа списка доступных рейсов;

- Form1 – является формой главного меню программы и служит для вызова функционала программы;

- Form2 – форма о программе, содержащая информацию о разработчике;

- Form4 – форма продажи билетов, служащая для ввода информации о проданном билете в базу данных;

- Form6 – форма статистики проданных билетов;

- Form7 – форма возврата билета, служащая для удаления информации о возвращенном или проданном билете;

- Form8 – форма для занесения данных о новом рейсе в базу данных;

- Form9 – форма календаря для просмотра текущего месяца и даты.

### **3.2 Проектирование и реализация интерфейса задачи**

При открытии программы вам предстают две формы – «Главное меню» и «Список Доступных Рейсов».

Форма «Главное меню» (рисунок 5) служит для вызова большей части функционала программы такого как: о программе, календарь, статистика продаж, редактор рейсов, а также список доступных рейсов, из которого вызывается функционал продажи авиабилетов.

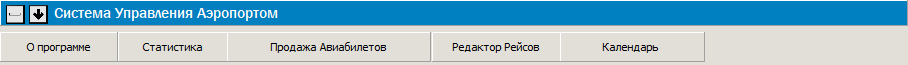


Рисунок 5 – Главное меню программы

Форма «Главное меню» содержит следующие компоненты Delphi:

* TPanel – служит для показа пунктов меню;
* TImage – один компонент служит как кнопка закрытия программы, а другой - как кнопка скрытия программы в фоновый режим.

Главной формой является «Список Доступных Рейсов» (рисунок 6).

Она первой отображается при инициализации проекта.

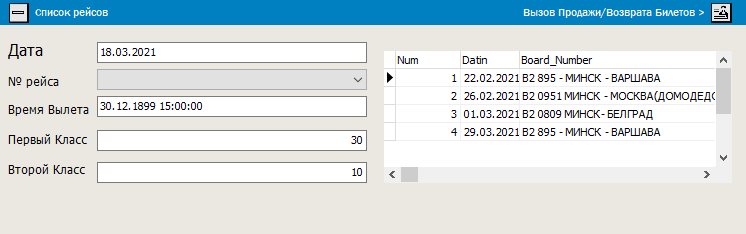


Рисунок 6 – Форма «Cписок Доступных Рейсов»

Основное назначение формы «Список Доступных Рейсов» – просмотр данных о доступных на сегодняшний день рейсах и доступ к продаже и возврату билета.

Форма содержит следующие компоненты Delphi:

- DBGrid – для показа данных о рейсах, занесенных в базу данных;

- TLabel – для обозначения полей;

- TEdit (текстовое поле) – для занесения текущей даты;

- TDBCombobox (комбинированное текстовое поле) - для показа доступных рейсов;

- TDBEdit - заполняющие значение на основе выбранного пункта в DBGrid и использующихся для просмотра данных о времени вылета и приблизительном количестве мест;

-TButton - для вызова формы продажи билетов

-TImage - кнопка закрытия формы.

В форме «Продажа билетов» (рисунок 7) Вы вносите информацию о номере рейса и дате продажи, а также выбирайте для какого класса бронирования **данный билет.**

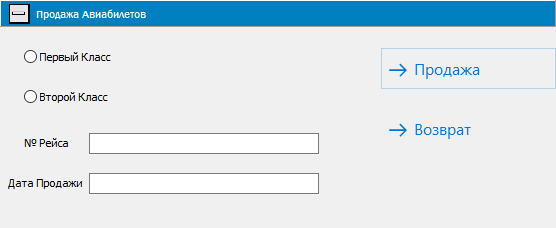


Рисунок 7 – Форма продажи билетов

Форма содержит следующие компоненты Delphi:

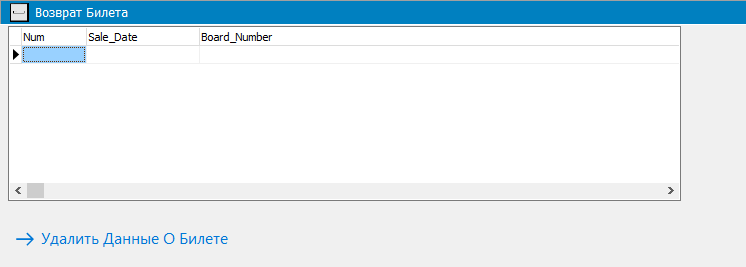
- TEdit (текстовое поле) – Для ввода номера рейса и даты продажи;

- TLabel – для обозначения полей;

-TButton – для занесения данных о продаже в базу данных и вызова формы возврата билета;

-TRadioButton – Для выбора класса бронирования.

**В форме «Возврат билета» (рисунок 8) Вы выбирайте в DBGrid тот билет, информацию о котором вы хотите удалить из базы данных.**

****

**Рисунок 8 – Форма Возврата Билета**

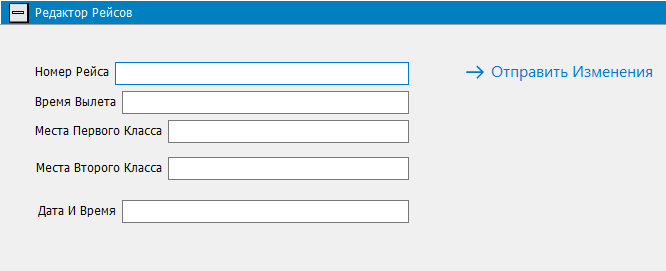
Форма содержит следующие компоненты Delphi:

- DBGrid – для показа данных о проданных билетах, занесенных в базу данных и выбора билета, информацию о котором надо удалить.

-TButton – для отправки запроса к базе данных о удалении информации.

**Теперь вернемся к компонентам главного меню программы.**

**Для вызова редактора рейсов щелкните на кнопку «Редактор рейсов» (рисунок 9) в главном меню.**

****

**Рисунок 9 – Форма «Редактора рейсов»**

В форме «Редактор рейса» Вы вносите данные о рейсе, который Вам надо добавить в предназначенных для этого полях, и отправляете запрос к базе данных о внесении данных при помощи кнопки «Отправить изменения».

Форма содержит следующие компоненты Delphi:

- TEdit (текстовое поле) – Для ввода номера рейса , времени вылета , количество мест первого и второго класса , а также даты вылета;

- TLabel – для обозначения полей;

-TButton – для отправки изменений в базу данных.

Также в программе присутствует просмотр статистики уже проданных билетов. Форма представлена на рисунке 10.

Форма содержит следующие компоненты Delphi:

- TDBGrid – показывающую статистику уже проданных билетов;

- TImage – один компонент служит как кнопка закрытия формы , второй как кнопка обновления информации на форме;

-TLabel- заголовок формы.

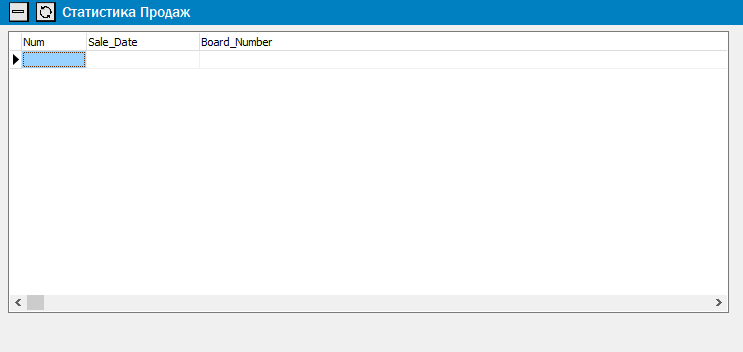


Рисунок 10 – Форма «Статистики»

В программе также присутсвует календарь вид которой представлен на рисунке 11.

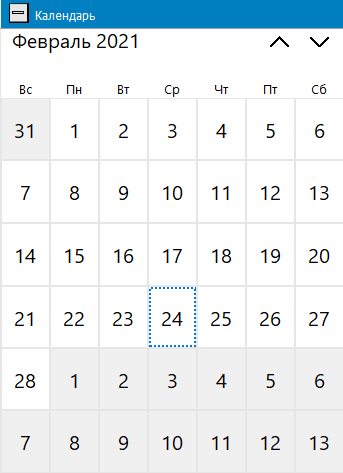


Рисунок 11 – Форма «Календаря»

Форма содержит следующие компоненты Delphi:

Calendarview – Для показа календаря

TPanel- Содержит заголовок формы и кнопку закрытия

-TImage - кнопка закрытия формы.

### **3.3 Разработка справочной системы**

Разработанная справочная система для обучающе-тестирующей программы по учебной дисциплине «Аэропорт» позволяет пользователю ознакомиться с краткой информацией о авторе программы и для чего она предназначена.

Cправка создана вручную без использования шаблона aboutbox.

Вид справки «О программе» представлен на рисунке 12.

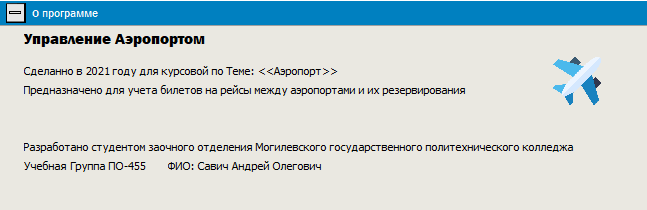


Рисунок 12 – Вид справки «О программе»

### **3.4 Тестирование задачи**

Тестирование проводилось с целью проверки корректности работы программы и её отладки.

Тестирование программы осуществлялось на персональном компьютере со следующими техническими характеристиками:

* Процессор – AMD FX(tm)-6350 Six-Core Processor 3.90 GHz
* Видеокарта – Nvidia Geforce GTX 750 TI
* Оперативная Память - 8,00 ГБ DDR3
* Операционная система – Windows Server 2019 Datacenter

Для обеспечения устойчивости программы и базы данных в форме продажи билетов и введены проверки на отсутствие выбора класс бронирования и отсутствие ввода данных а также проверка при выборе класса бронирования но отсутствии ввода данных номера рейса и даты продажи

В форме редактора рейсов введены проверки на ввод данных и частичный ввод данных.

Допустим мы не введем данные в форму продажи билетов – программа выдаст следующее сообщение изображенное на рисунке 13.

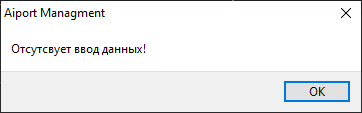


Рисунок 13 – Ошибка о отсутствии ввода

Если отсутствует выбор в одной из двух кнопок RadioButton программа выдаст следующее сообщение изображенное на рисунке 14.

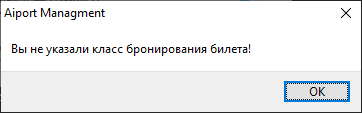


Рисунок 14 – Ошибка об отсутствии выбора класса бронирования

При выборе класса бронирования но отсутствии ввода данных о номере рейса и дате продажи будет выведено следующее сообщение изображенное на рисунке 15.

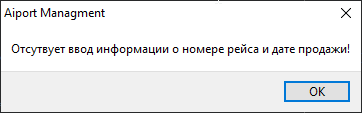


Рисунок 15 – Ошибка об отсутствии ввода данных о номере рейса и дате продажи

Если в редакторе рейсов не ввести данные то выйдет ошибка такого же вида как и на рисунке 13.

Если в редакторе рейсов отсутствует ввод части данных то выйдет следущая ошибка изображенная на рисунке 16.

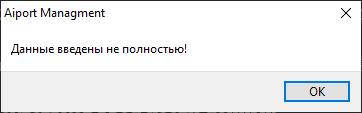


Рисунок 16 – Ошибка об отсутствии части данных

При тестировании программы в Delphi получены те же результаты, что и ожидались.

Программа устойчива и работоспособна.

### **4 Мероприятия по энерго- и ресурсосбережению**

В современном мире, в эпоху развивающейся промышленности необходимо принимать меры для сохранения окружающей нас среды. Поэтому в Республике Беларусь, как и в любой другой развитой стране, принята политика по ресурсо- и энергосбережению. Данная политика затрагивает все сферы общественной жизни, и конечно, не обошла и сферу образования. Стандарт управления энергопотреблением компьютеров, описывает три различных режима работы компьютера. Эти режимы отличаются потребляемой мощностью электроэнергии. Ждущий режим - позволяет экономить энергию, за счет отключения всех периферийных устройств, кроме ЦПУ, ОЗУ и материнской платы. Спящий режим - позволяет экономить электроэнергию на 100%, так как отключаются все периферийные устройства. Основной режим - при этом режиме экономии нет. Потребляемая мощность около 400Ватт.

Для нахождения количества рабочих дней, в течение которых разрабатывался программный продукт, используем формулу:

n = Тпк /(8-tрп), (1)

где Тпк – время работы компьютера, ч;

Тпк =91,05 ч.

tрп – суммарное время регламентированных перерывов, в течение

рабочего дня, ч;

tрп =50 мин=5/6 часа.

дн

Для нахождения суммарной продолжительности регламентированных перерывов в течение всего времени разработки программного модуля:

Трп = n × tрп,(2)

где Трп – суммарная продолжительность регламентированных перерывов в течение всего времени разработки программного модуля;



Стоимость сэкономленной электроэнергии рассчитывается по формуле:

Сэн=Трп×(Wпк-Wсп)× Сэ, (3)

где Wпк – потребляемая мощность ПК, кВт;

Wсп – потребляемая мощность компьютера в ждущем режиме;

Сэ – стоимость 1 кВт электроэнергии, руб.

Wпк = 0,55 кВт;

Wсп = 0,35 кВт;

Сэ=0,30 руб/Кв;



### **Заключение**

В результате выполнения курсового проекта была разработана программа для оперативного ведения продажи авиабилетов в день вылета и их возврат. Кроме этого были расширены знания из данной предметной области и укреплены знания объектно-ориентированного языка программирования Delphi. При реализации проекта были пройдены этапы описания и постановки задачи продажи авиабилетов, кодирования программы на алгоритмический язык и тестирования полученного приложения.

По окончании выполнения проекта закреплены знания по языку программирования, научился работать с новыми компонентами Delphi.

### **Список используемых источников**

1. ГОСТ ИСО/ МЭК 2382-99. Информационные технологии. Словарь. Ч. 1. Основные термины.
2. ГОСТ19.701-90 (ИСО 5807-85). Описание символов. Правила применения символов и выполнения схем.
3. СТУ 01-32-2019. Стандарт учреждения. Общие требования к оформлению текстовых документов».
4. Архангельский, А.Я. Программирование в среде Delphi 7 / А.Я. Архангельский – Москва: Бином, 2003.
5. Баженова, И.Ю. Самоучитель программиста Delphi 7 / И.Ю.Баженова – Москва: КУДИЦ ОБРАЗ, 2003.
6. Бобровский, С.И. Delphi 7. Учебный курс/ С.И.Бобровский - Санкт-Петербург: Питер, 2004.
7. Васильков, Ю.В., Василькова, Н.Н. Компьютерные технологии вычислений в математическом моделировании / Ю.В.Васильков, Н.Н.Василькова. – Москва: Финансы и статистика, 2004.-255 с.
8. Вержбицкий, В.М. Численные методы (линейная алгебра и нелинейные уравнения): Учеб. пособие для вузов / В.М.Вержбицкий. – Москва: Высш.шк., 2000. – 266 с.
9. Галисеев, Г.В. Программирование в среде Delphi 7. Самоучитель/ Г.В.Галисеев.-Москва: Издательский дом «Вильямс», 2003.-288 с.
10. Фаронов, В.В Программирование баз данных в Delphi 7 / В.В. Фаронов – Санкт-Петербург: Питер, 2006.
11. Шпак, Ю.А. Delphi 7 на примерах / Ю.А.Шпак – Украина: Юниор, 2003.

# Приложение А

(обязательное)

Текст программы

unit InfoReisy;

interface

uses

Windows, Messages, SysUtils, Variants, Classes, Graphics,

Controls, Forms, Dialogs, Menus, StdCtrls, DB,

ADODB, DBCtrls, Vcl.Mask, Vcl.ExtCtrls, System.Win.TaskbarCore, Vcl.Taskbar,

Vcl.Imaging.pngimage, Vcl.Grids, Vcl.DBGrids,OknoProdazhi;

type

TForm5 = class(TForm)

Label2: TLabel;

edDate: TEdit;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

ADOTable1: TADOTable;

ADOQuery1: TADOQuery;

ADOTable1Num: TAutoIncField;

ADOTable1Datin: TDateTimeField;

ADOTable1Board\_Number: TWideStringField;

ADOTable1Depart\_Time: TDateTimeField;

ADOTable1First\_Class: TIntegerField;

ADOTable1Second\_Class: TIntegerField;

dbeTime: TDBEdit;

dbeClass1: TDBEdit;

dbeClass2: TDBEdit;

Panel1: TPanel;

Image1: TImage;

DataSource1: TDataSource;

DBGrid1: TDBGrid;

cmbNumber: TDBComboBox;

ADOConnection1: TADOConnection;

bttProdaga: TImage;

Label1: TLabel;

Label7: TLabel;

procedure FormCreate(Sender: TObject);

procedure cmbNumberDropDown(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Image1Click(Sender: TObject);

procedure bttProdagaClick(Sender: TObject);

procedure Label7MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form5: TForm5;

implementation

{$R \*.dfm}

procedure TForm5.FormCreate(Sender: TObject);

var d: tdatetime;

begin

d:=date;

edDate.Text:=datetostr(d);

ADOTable1.Open;

end;

procedure TForm5.Image1Click(Sender: TObject);

begin

Form5.Hide;

end;

procedure TForm5.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm5.Label7MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm5.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm5.bttProdagaClick(Sender: TObject);

begin

Form4.Show;

end;

procedure TForm5.Button1Click(Sender: TObject);

begin

Close;

end;

procedure TForm5.cmbNumberDropDown(Sender: TObject);

begin

ADOTable1.Open;

ADOTable1.First;

while not ADOTable1.Eof do

begin

cmbNumber.Items.Add(AdoTable1.FieldByName('Board\_Number').AsString);

ADOTable1.Next;

end;

end;

end.

unit AboutProgram;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.pngimage,

Vcl.ExtCtrls;

type

TForm2 = class(TForm)

Image1: TImage;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Label6: TLabel;

Panel1: TPanel;

Image2: TImage;

Label7: TLabel;

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Image2Click(Sender: TObject);

procedure Label7MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form2: TForm2;

implementation

{$R \*.dfm}

procedure TForm2.Image2Click(Sender: TObject);

begin

Form2.Close;

end;

procedure TForm2.Label7MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm2.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.

unit Calendar;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.pngimage,

Vcl.ExtCtrls, Vcl.WinXCalendars;

type

TForm9 = class(TForm)

CalendarView1: TCalendarView;

Panel1: TPanel;

Image1: TImage;

Label1: TLabel;

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Image1Click(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form9: TForm9;

implementation

{$R \*.dfm}

procedure TForm9.Image1Click(Sender: TObject);

begin

Form9.Close;

end;

procedure TForm9.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm9.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.

unit EditFlights;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Imaging.pngimage,

Vcl.ExtCtrls, Data.DB, Data.Win.ADODB;

type

TForm8 = class(TForm)

Panel1: TPanel;

Image1: TImage;

Label1: TLabel;

Label2: TLabel;

Edit1: TEdit;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

Edit2: TEdit;

Edit3: TEdit;

Edit4: TEdit;

Edit5: TEdit;

Label6: TLabel;

Button1: TButton;

ADOConnection1: TADOConnection;

ADOTable1: TADOTable;

ADOTable1Num: TAutoIncField;

ADOTable1Datin: TDateTimeField;

ADOTable1Board\_Number: TWideStringField;

ADOTable1Depart\_Time: TDateTimeField;

ADOTable1First\_Class: TIntegerField;

ADOTable1Second\_Class: TIntegerField;

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Image1Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form8: TForm8;

implementation

{$R \*.dfm}

procedure TForm8.Button1Click(Sender: TObject);

begin

if (Trim(Edit1.Text) <> '') and (Trim(Edit2.Text) <> '') and (Trim(Edit3.Text) <> '')and (Trim(Edit5.Text) <> '') then begin

ADOTable1.Open;

ADOTable1.Insert;

ADOTable1.FieldByName('Board\_Number').AsString:= Edit1.Text;

ADOTable1.FieldByName('Depart\_Time').AsString:= Edit2.Text;

ADOTable1.FieldByName('First\_Class').AsString:= Edit3.Text;

ADOTable1.FieldByName('Second\_Class').AsString:= Edit4.Text;

ADOTable1.FieldByName('Datin').AsString:= Edit5.Text;

ADOTable1.Post;

end;

if (Edit1.Text = '') and (Edit2.Text = '') and (Edit3.Text = '') and (Edit4.Text = '') and (Edit5.Text = '') then begin

showmessage('Отсутствует Ввод Данных!');

end;

if (Edit1.Text = '')then begin

showmessage('Данные введены не полностью!');

end;

if (Edit1.Text = '') and (Edit2.Text = '') then begin

showmessage('Данные введены не полностью!');

end;

if (Edit1.Text = '') and (Edit2.Text = '') and (Edit3.Text = '') then begin

showmessage('Данные введены не полностью!');

end;

if (Edit1.Text = '') and (Edit2.Text = '') and (Edit3.Text = '') and (Edit4.Text = '') then begin

showmessage('Данные введены не полностью!');

end;

end;

procedure TForm8.FormCreate(Sender: TObject);

begin

ADOTable1.Open;

end;

procedure TForm8.Image1Click(Sender: TObject);

begin

Form8.Close;

end;

procedure TForm8.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm8.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.

unit Mainmenu;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.pngimage,

Vcl.StdCtrls,AboutProgram,Statistika,EditFlights,Calendar, Vcl.WinXPickers;

type

TForm1 = class(TForm)

Panel1: TPanel;

Label1: TLabel;

Panel3: TPanel;

Panel4: TPanel;

Panel5: TPanel;

Panel7: TPanel;

Image1: TImage;

Panel6: TPanel;

Panel8: TPanel;

Image2: TImage;

procedure Panel3Click(Sender: TObject);

procedure Panel6Click(Sender: TObject);

procedure Panel7Click(Sender: TObject);

procedure Panel5Click(Sender: TObject);

procedure Panel4Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Button1Click(Sender: TObject);

procedure Label1DblClick(Sender: TObject);

procedure Image2Click(Sender: TObject);

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

implementation

{$R \*.dfm}

uses InfoReisy;

procedure TForm1.Button1Click(Sender: TObject);

begin

Form2.Show;

Form8.Show;

Form6.Show;

Form9.Show;

Form5.Show;

end;

procedure TForm1.Image1Click(Sender: TObject);

begin

Application.Terminate;

end;

procedure TForm1.Image2Click(Sender: TObject);

begin

Application.Minimize;

end;

procedure TForm1.Label1DblClick(Sender: TObject);

begin

Panel8.Visible:=True;

end;

procedure TForm1.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm1.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm1.Panel3Click(Sender: TObject);

begin

form2.show;

end;

procedure TForm1.Panel4Click(Sender: TObject);

begin

Form8.Show;

end;

procedure TForm1.Panel5Click(Sender: TObject);

begin

Form6.Repaint;

Form6.Show;

Form6.Refresh;

end;

procedure TForm1.Panel6Click(Sender: TObject);

begin

Form9.Show;

end;

procedure TForm1.Panel7Click(Sender: TObject);

begin

Form5.Show;

end;

end.

unit OknoProdazhi;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.StdCtrls, Data.DB,

Data.Win.ADODB, Vcl.Imaging.pngimage,ReturnBilet;

type

TForm4 = class(TForm)

Panel1: TPanel;

RadioButton1: TRadioButton;

RadioButton2: TRadioButton;

Button1: TButton;

Button2: TButton;

ADOConnection1: TADOConnection;

ADOTable1: TADOTable;

Label1: TLabel;

Edit1: TEdit;

Label2: TLabel;

Edit2: TEdit;

Image1: TImage;

Label3: TLabel;

ADOTable1Num: TAutoIncField;

ADOTable1Sale\_Date: TDateTimeField;

ADOTable1Board\_Number: TWideStringField;

ADOTable1Class: TWideStringField;

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Button1Click(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Button2Click(Sender: TObject);

procedure Label3MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form4: TForm4;

implementation

{$R \*.dfm}

procedure TForm4.Button1Click(Sender: TObject);

begin

if (RadioButton1.Checked=true) and (Trim(Edit1.Text) <> '') and (Trim(Edit2.Text) <> '') then begin

ADOTable1.Open;

ADOTable1.Insert;

ADOTable1.FieldByName('Class').AsString:= 'Первый Класс';

ADOTable1.FieldByName('Board\_Number').AsString:=Edit1.Text;

ADOTable1.FieldByName('Sale\_Date').AsString:=Edit2.Text;

ADOTable1.Post;

end;

if (RadioButton2.Checked=true) and (Trim(Edit1.Text) <> '') and (Trim(Edit2.Text) <> '') then begin

ADOTable1.Open;

ADOTable1.Insert;

ADOTable1.FieldByName('Class').AsString:= 'Второй Класс';

ADOTable1.FieldByName('Board\_Number').AsString:=Edit1.Text;

ADOTable1.FieldByName('Sale\_Date').AsString:=Edit2.Text;

ADOTable1.Post;

end;

if (RadioButton1.Checked=false) and (RadioButton2.Checked=false ) and (Edit1.Text = '') and (Edit2.Text = '') then begin

showmessage('Отсутсвует ввод данных!');

end;

if (RadioButton1.Checked=false) and (RadioButton2.Checked=false ) and (Trim(Edit1.Text) <> '') and (Trim(Edit2.Text) <> '') then begin

showmessage('Вы не указали класс бронирования билета!');

end;

if (RadioButton1.Checked=true) and (Edit1.Text = '') and (Edit2.Text = '') then begin

showmessage('Отсутвует ввод информации о номере рейса и дате продажи!');

end;

if (RadioButton2.Checked=true) and (Edit1.Text = '') and (Edit2.Text = '') then begin

showmessage('Отсутвует ввод информации о номере рейса и дате продажи!');

end;

end;

procedure TForm4.Button2Click(Sender: TObject);

begin

Form7.Show;

end;

procedure TForm4.FormCreate(Sender: TObject);

begin

ADOTable1.Open;

end;

procedure TForm4.Image1Click(Sender: TObject);

begin

Form4.Close;

end;

procedure TForm4.Label3MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm4.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.

unit ReturnBilet;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtCtrls, Vcl.Imaging.pngimage,

Vcl.StdCtrls, Data.DB, Data.Win.ADODB, Vcl.Grids, Vcl.DBGrids;

type

TForm7 = class(TForm)

Panel1: TPanel;

Label1: TLabel;

Image1: TImage;

DBGrid1: TDBGrid;

ADOConnection1: TADOConnection;

ADOTable1: TADOTable;

DataSource1: TDataSource;

Label2: TLabel;

Button1: TButton;

ADOTable1Num: TAutoIncField;

ADOTable1Sale\_Date: TDateTimeField;

ADOTable1Board\_Number: TWideStringField;

ADOTable1Class: TWideStringField;

procedure Image1Click(Sender: TObject);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Button1Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form7: TForm7;

implementation

{$R \*.dfm}

procedure TForm7.Button1Click(Sender: TObject);

begin

ADOTable1.Open;

ADOTable1.Delete;

end;

procedure TForm7.FormCreate(Sender: TObject);

begin

ADOTable1.Open;

end;

procedure TForm7.Image1Click(Sender: TObject);

begin

Form7.Close;

end;

procedure TForm7.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm7.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.

unit Statistika;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Data.DB, Vcl.Grids, Vcl.DBGrids,

Vcl.ExtCtrls, Vcl.StdCtrls, Data.Win.ADODB, Vcl.Imaging.pngimage;

type

TForm6 = class(TForm)

Panel1: TPanel;

DBGrid1: TDBGrid;

DataSource1: TDataSource;

ADOConnection1: TADOConnection;

ADOTable1: TADOTable;

Label1: TLabel;

Image1: TImage;

Image2: TImage;

ADOTable1Num: TAutoIncField;

ADOTable1Sale\_Date: TDateTimeField;

ADOTable1Board\_Number: TWideStringField;

ADOTable1Class: TWideStringField;

procedure FormCreate(Sender: TObject);

procedure Image1Click(Sender: TObject);

procedure Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

procedure Image2Click(Sender: TObject);

procedure Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form6: TForm6;

implementation

{$R \*.dfm}

procedure TForm6.FormCreate(Sender: TObject);

begin

ADOTable1.Open;

end;

procedure TForm6.Image1Click(Sender: TObject);

begin

DBGrid1.DataSource.DataSet.Refresh;

Form6.Close;

end;

procedure TForm6.Image2Click(Sender: TObject);

begin

Form6.Refresh;

Form6.Repaint;

ADOTable1.Open;

ADOTable1.Refresh;

DBGrid1.DataSource.DataSet.Refresh;

DBGrid1.Refresh;

DBGrid1.Repaint;

end;

procedure TForm6.Label1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

procedure TForm6.Panel1MouseDown(Sender: TObject; Button: TMouseButton;

Shift: TShiftState; X, Y: Integer);

begin

ReleaseCapture;

Perform(WM\_SysCommand,$F012,0);

end;

end.