



Государственное бюджетное образовательное учреждение высшего образования
Московской области

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ

Колледж космического машиностроения и технологии

КУРСОВОЙ ПРОЕКТ

По МДК.01.02 «Прикладное программирование»

Тема: «Разработка приложения “Предприятие быстрого питания”»

Выполнил студент

Попкова Алена Игоревна

Группа П1-17

_____ (подпись)

Проверил преподаватель

Гусятинер Леонид Борисович

_____ (подпись)

_____ (Дата сдачи работы)

Королев, 2020

Содержание	
Введение	3
Глава 1. Теоретическая часть	4
1.1. Описание предметной области «Предприятие быстрого питания»	4
1.1.1. История быстрого питания.....	4
1.1.2. Причины популярности.....	4
1.1.3. Классификация предприятий	5
1.2. Описание существующих разработок.....	6
Глава 2. Проектная часть	8
2.1. Диаграмма прецедентов.....	8
2.2. Выбор инструментов.....	10
2.3. Проектирование сценария.....	11
2.4. Построение диаграммы классов	12
2.5. Описание главного модуля	12
2.6. Описание спецификаций к модулям	14
2.7. Описание модулей.....	17
2.8. Описание тестовых наборов модулей	20
2.9. Описание применения средств отладки	25
2.10. Анализ оптимальности использования памяти и быстродействия.....	28
Глава 3. Эксплуатационная часть.....	30
3.1. Руководство оператора	30
3.1.1. Назначение программы	30
3.1.2. Условия выполнения программы	30
3.1.3. Выполнение программы	31
3.1.4. Сообщение оператору	34
Заключение	35
Список литературы и интернет-источников.....	36
Приложение 1. Модуль AboutProgramm.....	37
Приложение 2. Модуль prodmenu	38
Приложение 3. Модуль finance.....	40
Приложение 4. Модуль данных dmd.....	42

Введение

Данный курсовой проект посвящен теме «Разработка приложения “Предприятие быстрого питания”». Актуальность данной темы можно обусловить тем, что в современном мире каждому предприятию быстрого питания нужно вести учет, будь это учет продуктового склада либо финансовый учет. Целью создания этого приложения было достигнуть удобного интерфейса для пользователя.

В первой главе описывается предметная область, которой посвящен данный проект, а также существующие разработки по выбранной теме.

Во второй главе описывается проектная часть, а именно диаграмма прецедентов, выбор инструментов, диаграмма классов, созданное приложение, тестирование и анализ оптимальности использования памяти и быстродействие приложения.

Третья глава является эксплуатационной частью, включающая в себя руководство оператора.

Последняя часть записки состоит из заключения, списка литературы и источников.

Глава 1. Теоретическая часть

1.1. Описание предметной области «Предприятие быстрого питания»

Быстрое питание — это питание с уменьшением времени приготовления и употребления пищи, вызванное острой нехваткой времени у современного общества.

1.1.1. История быстрого питания

Фаст-Фуд возник благодаря компании «White House» в 1921-ом году в Америке. Их фирменным блюдом стал гамбургер, что вызвала высокую заинтересованность у покупателей. Стоило такое блюдо 5 центов до 1946 года. Владелец компании Билли Инграм придумал своеобразную рекламу своих продуктов. Специально нанятые люди в белых халатах создавали впечатление, что даже врачи покупают гамбургеры.

В 1948 году у «White House» появился серьезный конкурент McDonald's.

Гонконгские чхачханьтхэны — азиатская разновидность ресторанов быстрого обслуживания, возникшая в 1950-х годах. В этих заведениях подают блюда, сочетающие в себе китайские и европейские черты, в частности, смесь чая с кофе и сгущённым молоком, суп с лапшой быстрого приготовления и ветчиной, а также разнообразные варианты сэндвичей.

1.1.2. Причины популярности

Преимуществом этих блюд является быстрота приготовления и употребления, дешевизна. Это находит отклик у потребителя, также такая пища очень технологична, что позволяет сетям фастфуд быстро развиваться.

Ресторан быстрого питания (фаст-фуд)—тип предприятия общественного питания, характеризующийся быстрым приготовлением пищи, а также минимальным, или отсутствующим обслуживанием посетителей официантом.

Среди наиболее распространенных за рубежом предприятий быстрого обслуживания можно выделить три следующих типа предприятий, работающих по технологическим схемам:

- «доготовка — комплектация — отпуск»;
- «хранение — разогрев — комплектация — отпуск»;
- «хранение — разогрев — отпуск».

1.1.3. Классификация предприятий

Предприятия общественного питания классифицируют:

- по степени централизации производства (с законченным производственным циклом — работающие на сырье, на полуфабрикатах, заготовочные и не имеющие производства — раздаточные);
- по характеру обслуживаемого контингента (с изменяющимся контингентом — общедоступные, с постоянным контингентом — пищеблоки при школах заводах и т.п.);
- по признаку специализации (комплексные, общего типа, специализированные по выпуску определенных видов продукции);
- по методу обслуживания (обслуживание официантами, самообслуживание);
- по наценочной категории и уровню обслуживания (люкс — рестораны; высшей — рестораны, кафе, бары; первой — рестораны, кафе, специализированные предприятия; второй — общедоступные столовые, кафе, специализированные предприятия, буфеты; третьей — столовые и буфеты, обслуживающие рабочих, служащих, студентов и учащихся).

В зависимости от времени функционирования предприятия общественного питания могут быть постоянно действующими и сезонными.

Предприятия подразделяются также на стационарные и передвижные.

В соответствии с ГОСТ 50762-95 «Общественное питание. Классификация предприятий» в составе индустрии питания выделены следующие типы предприятий: ресторан, бар, кафе, столовая, закусочная. В

составе указанных типов в последние годы большую популярность завоевали узкоспециализированные предприятия быстрого обслуживания.

1.2. Описание существующих разработок

1С:Предприятие 8. Фастфуд. Фронт-офис

Главной целью "1С:Фастфуд" является автоматизация предприятий общественного питания, работающих в формате быстрого обслуживания.

С помощью данного решения могут быть автоматизированы все виды предприятий быстрого питания.

Главным достоинством, по моему мнению, является удобный интерфейс. Для быстрого обслуживания клиента персоналом это играет большую роль. Также данное решение позволяет настраивать интерфейс для конкретного пользователя.

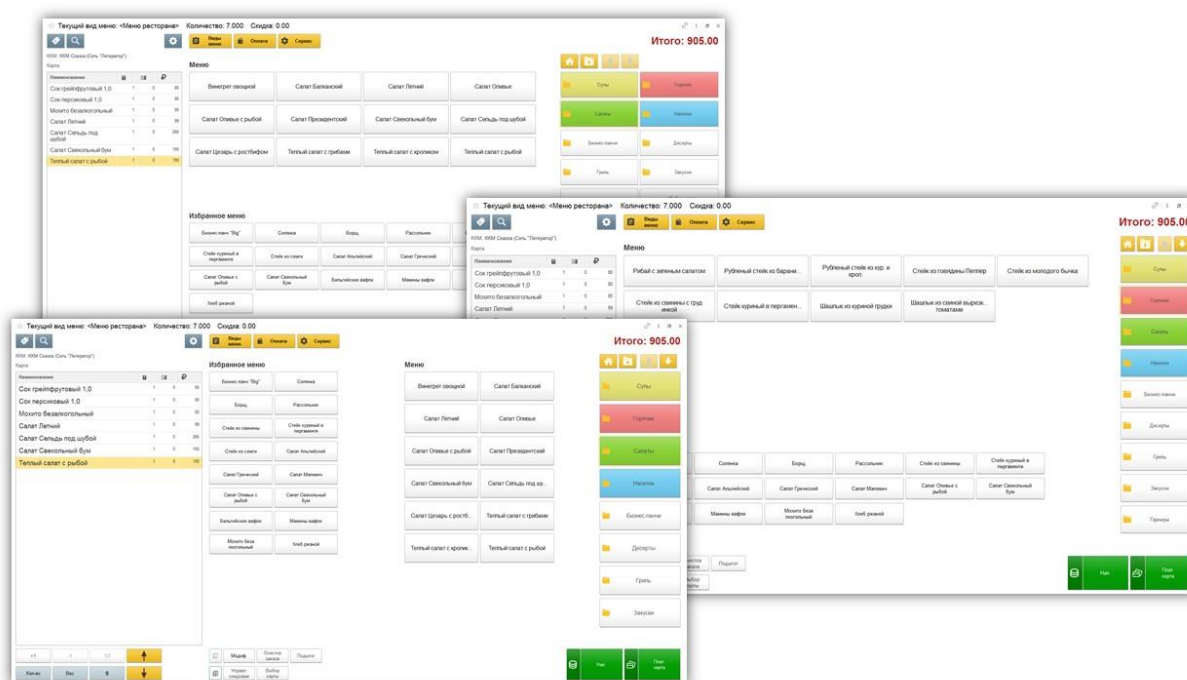


Рисунок 1. Настраиваемый интерфейс

Еще следует отметить, что наименование позиции в меню может отличаться от наименования товара/блюда в справочнике Номенклатура, что позволяет выводить на экран кассирам и на печать оптимальную информацию.

Агуша вода дет. питьевая 0,33л (Меню) (1С:Предприятие)

Записать и закрыть

Еще ?

Наименование: Агуша вода дет. питьевая Код: 0000001553

Входит в группу: Агуша Не в группе: 1

Номенклатура меню Фронт-офис

Номенклатура: Агуша вода дет. питьевая 0,33л

Рисунок 2. Меню

По моему мнению, данное приложение полностью справляется с выполнением своей главной задачи - автоматизацией предприятий быстрого питания.

Официальный сайт 1С:Предприятие 8. Фастфуд. Фронт-офис – [1]

Глава 2. Проектная часть

2.1. Диаграмма прецедентов

В ходе выполнения курсовой работы была построена диаграмма прецедентов для сотрудников предприятия быстрого питания (Рис.3). Данная диаграмма описывает работу сотрудников предприятия.

Главной задачей кассира является быстрое обслуживание клиентов, что включает в себя оформление заказа, прием денежных средств и выдача сдачи и т.д.

Директор по хозяйству отвечает за склад, а именно за ведение склада хозяйственных товаров и спец.одежды.

Бухгалтер должен выполнять свою главную задачу – введение бухгалтерского учета.

Шеф-повар прежде всего заведует производством, сюда входит контроль технологии производства пищи, составление меню заведения и составление заявок для заказа на необходимые продукты для производства

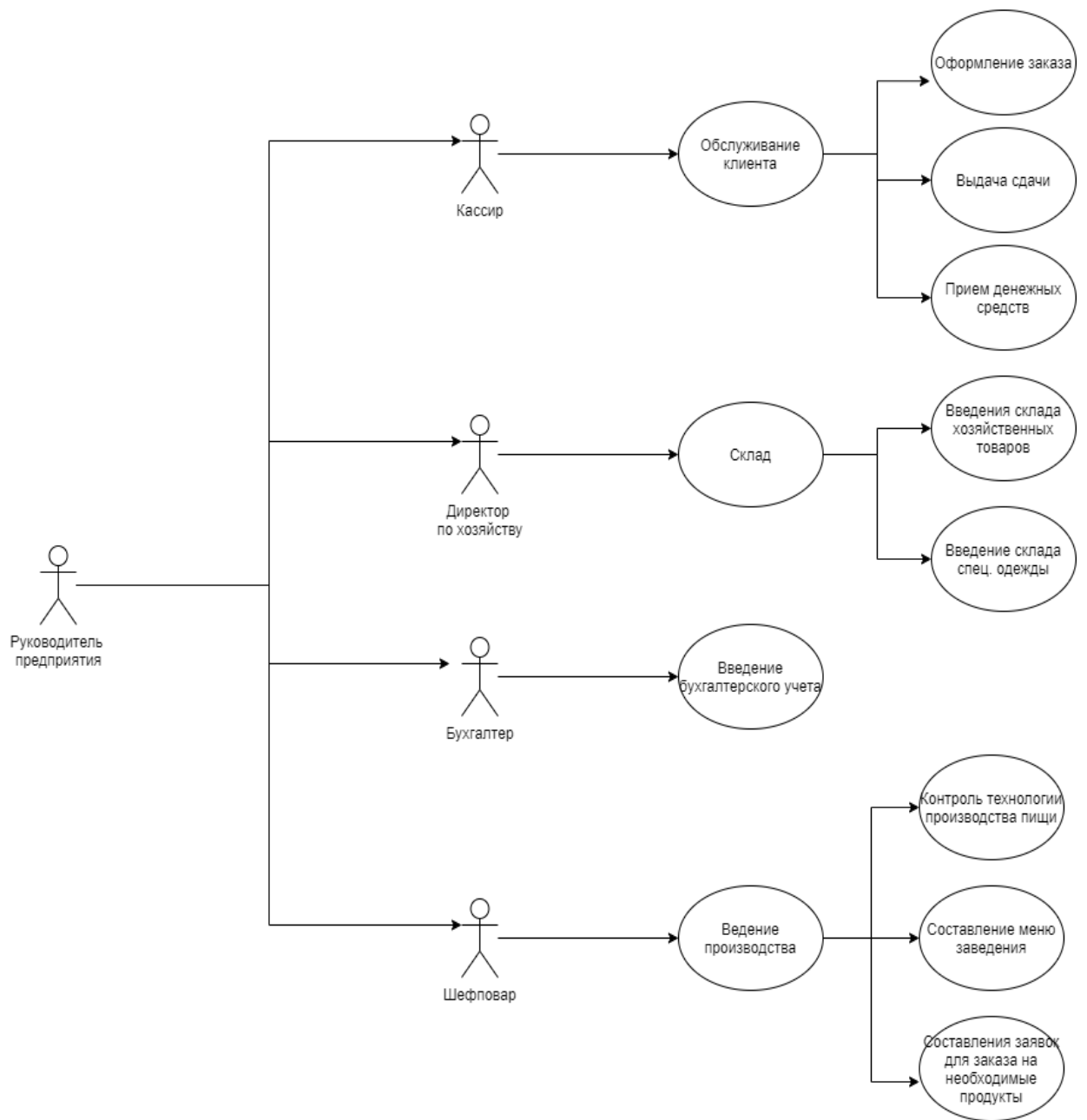


Рисунок 3. Диаграмма прецедентов

2.2. Выбор инструментов

При выборе инструментов учитывались критерии, приведенные в таблице ниже.

Важность критерия были определены для себя как слабая, средняя и высокая.

Таблица 1. Важность критерия оценивания.

Критерий	Участие в корпоративном проекте	Понятный синтаксис	Доступные IDE	Наличие официальной документации на русском языке	Минимальные затраты времени на освоение IDE и создание прикладных приложений.	Кол-во библиотек
Важность критерия	Высокая	Средняя	Высокая	Слабая	Высокая	Средняя

Сравнения двух, выбранных мною, языков программирования приведены в таблице ниже.

Таблица 2. Сравнение двух языков программирования

Критерий/ язык программирования	Java	Object Pascal
Участие в корпоративном проекте	+	+
Понятный синтаксис;	-	+
Доступные IDE;	-	+
Наличие официальной документации на русском языке	-	+
Минимальные затраты времени на освоение IDE и создание прикладных приложений.	+	+
Кол-во библиотек	+	-

По результатам сравнения для разработки данного курсового проекта был выбран Object Pascal.

Также была выбрана среда программирования Lazarus. Одним из главных преимуществ данной среды является кроссплатформенность. Мощность и быстрота также играют большую роль. Lazarus IDE совместим

для работы с большими проектами. Так же он имеет графический конструктор, что позволяет создавать формы за считанные минуты. Еще одним немаловажным преимуществом является доступность. Lazarus можно скачать бесплатно с официального сайта [2]

Для работы с базами данных мною была выбрана СУБД FireBird. FireBird полностью бесплатна. Также она является кроссплатформенной системой.

Официальный сайт FireBird [3]

2.3. Проектирование сценария

Разработанное мной приложение ориентированно на сценарий, приведенный на рисунке ниже (Рис.4).

Главным окном является «Меню». Это первое окно при открытии приложения. Оно содержит в себе три кнопки «Меню предприятия», «Финансы» и «Склад». Нажимая на определенную кнопку, открывается соответствующее окно. После открытия пользователь может начать работу с таблицей.

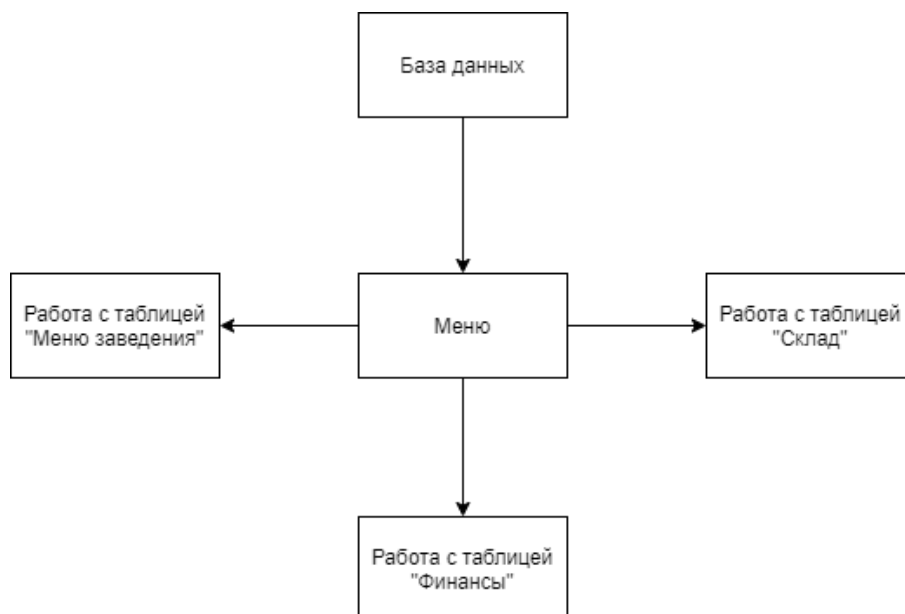


Рисунок 4. Сценарий приложения

2.4. Построение диаграммы классов

В приведенной ниже диаграмме классов изображены атрибуты и операции классов. Данная диаграмма является временной и в дальнейшем ходе разработки приложения будет дорабатываться.

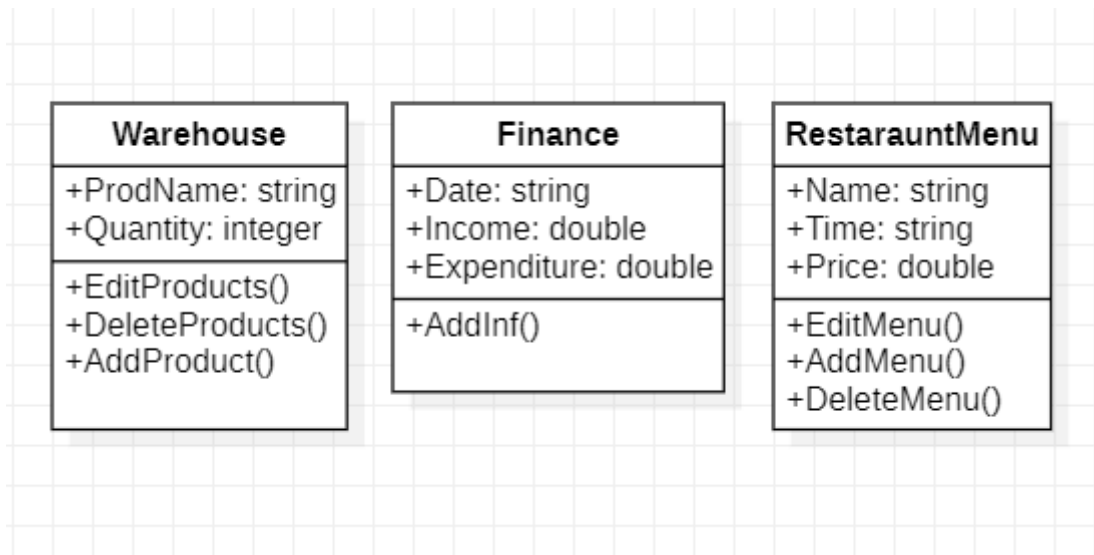


Рисунок 5. Диаграмма классов

Класс “Warehouse” содержит в себе следующие поля: ProdName (Наименование продукта) и Quantity (Количество), также имеет методы EditProducts() – изменение записей таблицы «Warehouse», DeleteProducts() – удаление записей таблицы, AddProducts() – добавление записей таблицы.

Класс “Finance” состоит из следующих полей: Date (Дата), Income (Доход) и Expenditure (Расход), а также имеет метод AddInf(), отвечающий за добавление информации в данную таблицу.

Класс “RestarauntMenu” содержит поля: Name (Наименование блюда), Time (Время приготовления) и Price (Цена), а также имеет методы EditMenu() – изменение записей таблицы «Warehouse», DeleteMenu() – удаление записей таблицы, AddMenu() – добавление записей таблицы.

2.5. Описание главного модуля

Главный модуль приложения выполняет создание экземпляров всех форм необходимых для работы приложения. Блок-схема показана на Рисунке

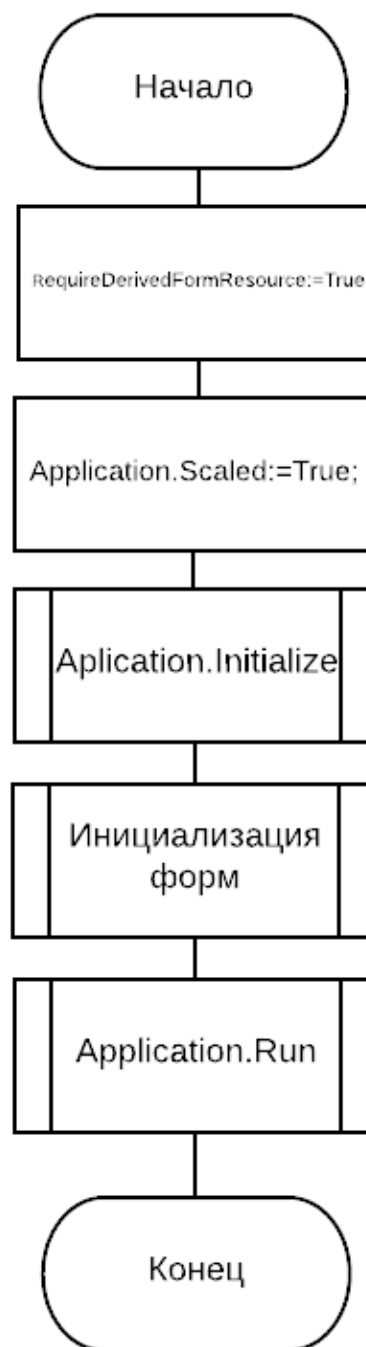


Рисунок 6. Блок-схема главного модуля

Листинг 2.5.1. Главный модуль.

```

program FastFood;

{$mode objfpc}{$H+}

uses
    {$IFDEF UNIX}{$IFDEF UseCThreads}
    cthreads,
    {$ENDIF}{$ENDIF}
    Interfaces, // this includes the LCL widgetset
    Forms, AppMenu, finance, prodmenu, warehouse, dmd, aboutprogramm;

{$R *.res}

begin
    RequireDerivedFormResource:=True;
    Application.Scaled:=True;
    Application.Initialize;
    Application.CreateForm(TAppMenu1, AppMenu1);
    //Создание экземпляра формы AppMenu
    Application.CreateForm(TFormFinance, FormFinance);
    //Создание экземпляра формы FormFinance
    Application.CreateForm(TFProdMenu, FProdMenu);
    //Создание экземпляра формы FProdMenu
    Application.CreateForm(TFWarehouse, FWarehouse);
    //Создание экземпляра формы FWarehouse
    Application.CreateForm(TDataModule2, DataModule2);
    //Создание экземпляра формы модуля данных DataModule2
    Application.CreateForm(TFAboutProgramm, FAboutProgramm);
    //Создание экземпляра формы FAboutProgramm
    Application.Run;
end.

```

2.6. Описание спецификаций к модулям

Класс формы «Финансы» состоит из полей: кнопка «Назад», компонент для отображения таблицы БД, поле ввода, надпись, невидимый компонент меню, пункт меню и методов обработки события Edit, создание формы и события кнопки «Назад».

Листинг 2.6.1. Класс формы «Финансы»

```
type
```

```

{ TFormFinance }

TFormFinance = class(TForm)           // Клас формы "Финансы"
    BackFromFinance: TButton;          // кнопка "Назад"
    DBGrid1: TDBGrid;                  // компонент для отображения
данных таблицы БД
    DBGrid2: TDBGrid;                  // компонент для отображения
данных таблицы БД
    dbnFinance: TDBNavigator;          // компонент для работы с
данными таблицы БД
    Edit1: TEdit;                      // поле ввода
    Label1: TLabel;                   // надпись
    MainMenu1: TMainMenu;              // невизуальный компонент меню
    MenuItem4: TMenuItem;              // пункт меню
    procedure BackFromFinanceClick(Sender: TObject); // процедура
события кнопки "Назад"
    procedure Edit1Change(Sender: TObject); // обработчик
события Edit
    procedure MenuItem4Click(Sender: TObject);
private
public

end;
```

Класс формы «Меню предприятия» состоит из полей: кнопка «Назад», компонент для отображения таблицы БД, поле ввода, надпись, невизуальный компонент меню, пункт меню и методов обработки события Edit, создание формы и события кнопки «Назад».

Листинг 2.6.2. Класс формы «Меню предприятия»

```

type

    { TFProdMenu }

    TFProdMenu = class(TForm)                                // Класс формы "Меню
    предприятия"                                              //
        ButMenuBack: TButton;                                // кнопка "Назад"
        DBGrid1: TDBGrid;                                    // компонент для отображения
        данных таблицы БД
        DBGrid2: TDBGrid;                                    // компонент для отображения
        данных таблицы БД
        dbnProdMenu: TDBNavigator;                            // компонент для работы с
        данными таблицы БД
        Edit1: TEdit;                                         // поле ввода
        Label1: TLabel;                                       // надпись
        MainMenu1: TMainMenu;                                // невизуальный компонент меню
        MenuItem4: TMenuItem;                                // пункт меню
        procedure ButMenuBackClick(Sender: TObject);         // процедура
        события кнопки "Назад"
        procedure Edit1Change(Sender: TObject);              // обработчик события
        Edit
        procedure MenuItem4Click(Sender: TObject);
    private
    public
    end;

```

Класс формы «Склад» состоит из полей: кнопка «Назад», компонент для отображения таблицы БД, поле ввода, надпись, невизуальный компонент меню, пункт меню и методов обработки события Edit, создание формы и события кнопки «Назад».

Листинг 2.6.3. Класс формы «Склад»

```

type

    { TFWarehouse }

    TFWarehouse = class(TForm)                                // Класс формы "Склад"
        BackFromWarehouse: TButton;                          // кнопка "Назад"
        DBGrid1: TDBGrid;                                    // компонент для отображения
        данных таблицы БД
        DBGrid2: TDBGrid;                                    // компонент для отображения
        данных таблицы БД
        dbnWarehouse: TDBNavigator;                          // компонент для работы с
        данными таблицы БД
        Edit1: TEdit;                                         // поле ввода
        Label1: TLabel;                                       // надпись

```



```

MainMenu1: TMainMenu;           // невидимый компонент меню
MenuItem4: TMenuItem;          // пункт меню
procedure BackFromWarehouseClick(Sender: TObject); // процедура
события кнопки "Назад"
procedure Edit1Change(Sender: TObject); // обработчик события
Edit
procedure MenuItem4Click(Sender: TObject);
private

public

end;

```

2.7. Описание модулей

В процессе работы над приложением были созданы модули выбора таблицы (AppMenu) и модули работы с таблицами (warehouse, prodmenu, finance). В модулях работы с таблицами были реализованы процедуры формирующие SQL – запрос поиска. Ниже приведен код одного из модуля (warehouse)

Листинг 2.7.1. Модуль Warehouse

```

unit warehouse;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Menus,
  DBGrids, DBCtrls;

type

  { TFWarehouse }

  TFWarehouse = class(TForm)                                // Форма "Склад"
    BackFromWarehouse: TButton;
    DBGrid1: TDBGrid;
    DBGrid2: TDBGrid;
    dbnWarehouse: TDBNavigator;
    Edit1: TEdit;
    Label1: TLabel;
    MainMenu1: TMainMenu;
    MenuItem4: TMenuItem;
    procedure BackFromWarehouseClick(Sender: TObject);
    procedure dbnWarehouseClick(Sender: TObject; Button:
TDBNavButtonType);
    procedure Edit1Change(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure MenuItem4Click(Sender: TObject);
  private
  public

  end;

var
  FWarehouse: TFWarehouse;

implementation
uses AppMenu, dmd, AboutProgramm;
{$R *.lfm}

{ TFWarehouse }

procedure TFWarehouse.BackFromWarehouseClick(Sender: TObject);
// Процедура события кнопки "Назад"
begin

```

```

        Close;
        AppMenu1.Show;
end;

procedure TFWarehouse.dbnWarehouseClick(Sender: TObject;
    Button: TDBNavButtonType);
begin

end;

procedure TFWarehouse.Edit1Change(Sender: TObject);
    // Процедура запроса поиска к таблице Warehouse
    var buff, strSearch: string;
begin
    buff := '%' + FWarehouse.Edit1.Text + '%';
    // Строка принимающая значение, введенное пользователем
    strSearch := QuotedStr(buff);
    // Подготовленная строка для запроса поиска
    with dmd.DataModule2.ibqWarehouse do
    begin
        Close;
        SQL.Clear; // Очищение предыдущего запроса
        SQL.Add('select * from WAREHOUSE where PROD_NAME like' +
strSearch + ' or PROD_QUAN like' + strSearch + ' order by PROD_NAME');
        // Запрос поиска
        open;
    end;
end;

procedure TFWarehouse.FormCreate(Sender: TObject);
begin

end;

procedure TFWarehouse.MenuItem4Click(Sender: TObject);
    // Процедура события кнопка меню "О программе"
begin
    FAboutProgramm.Show;
end;

```

На рисунке ниже приведен модуль данных, отвечающий за работу с БД. Он содержит в себе компонент транзакции TIBTransaction (ibtr), компонент соединения с базой данных TIBDataBase (ibdb), компоненты таблиц TIBTable (ibWarehouse, ibFinance, ibProdmenu) и компоненты TDataSource, отвечающие за управление потоками данных, за связи между набором данных и

визуальными компонентами. (dtsWarehouse, dtsFinance, dtsProdMenu, dtsWarehouseQ, dtsFinanceQ, dtsProdMenuQ), а также компоненты TIBQuery(ibqWarehouse, ibqFinance, ibqProdmenu), выполняющие все стандартные функции компонента запроса, которые в моем приложении применяются для поиска информации в таблицах

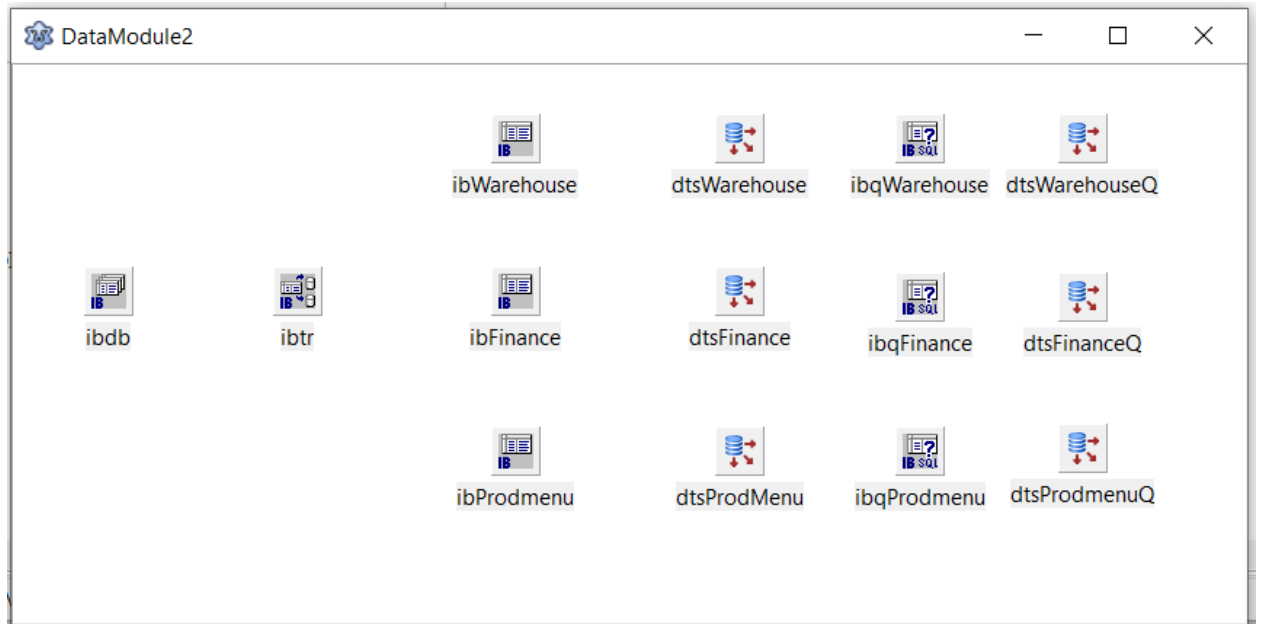


Рисунок 7. Модуль данных

2.8. Описание тестовых наборов модулей

На рисунках ниже приведены результаты работы добавления записи в таблице. Первое окно вывода таблицы предназначено для работы с таблицей, второе для поиска данных, при этом все данные отсортированы по алфавиту.

Таблица до добавления новой записи показана на рисунке 8

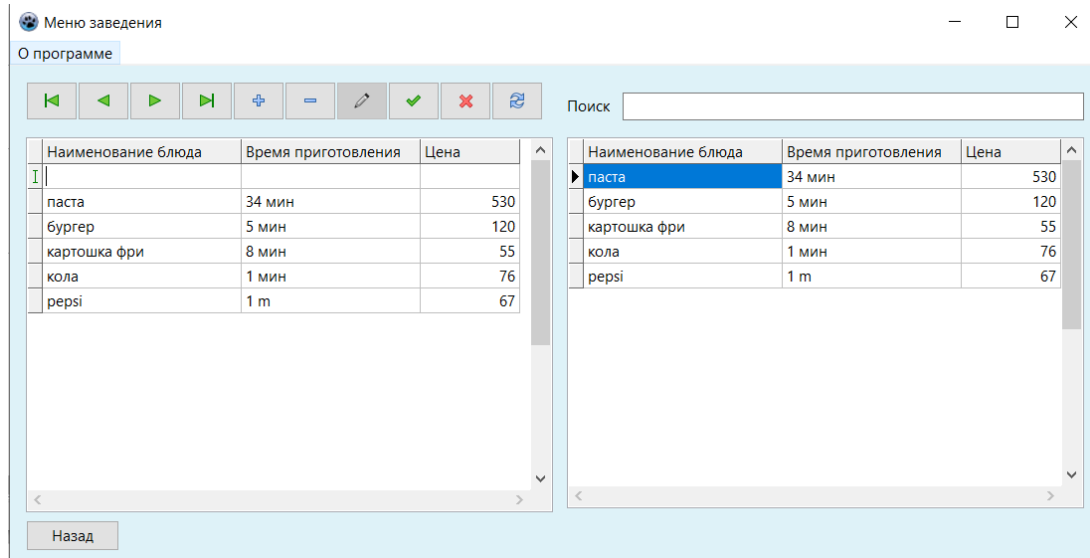


Рисунок 8. Таблица до добавления

Результат добавления новой записи показан ниже на рисунке 9

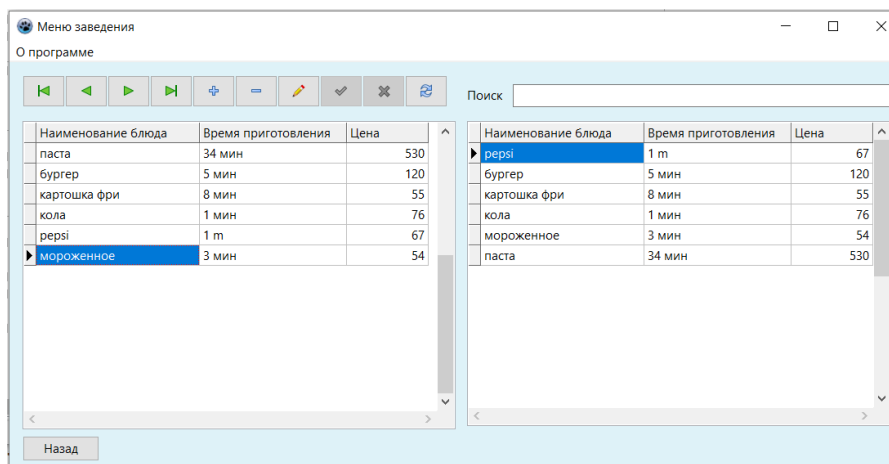


Рисунок 9. Таблица после добавления

В процессе тестирования было обнаружено следующее. При незаполнении поля со свойством Not Null выходит окно с системной ошибкой. В следующей версии приложения планируется разработать обработку данных ошибок и выводить предупреждения самим приложением.

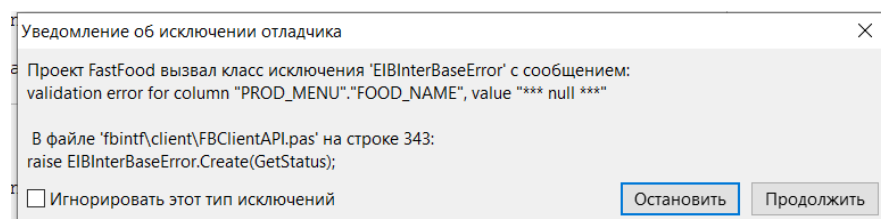
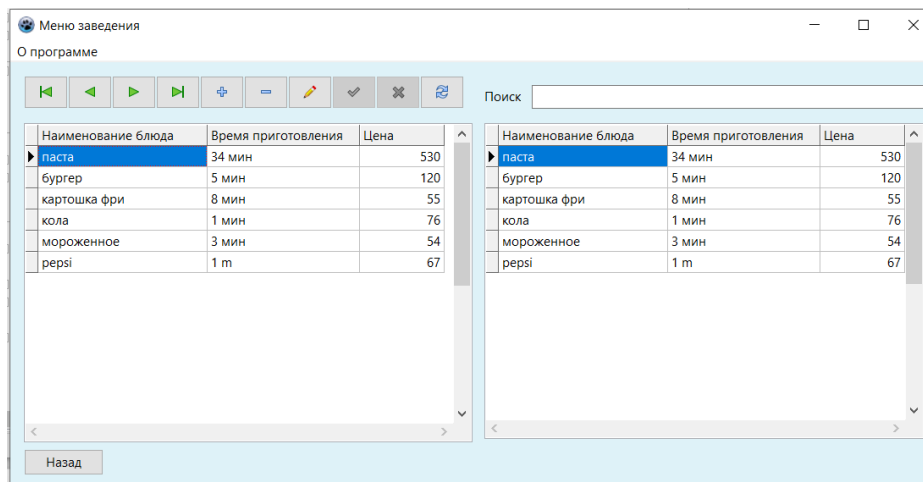


Рисунок 10. Системная ошибка

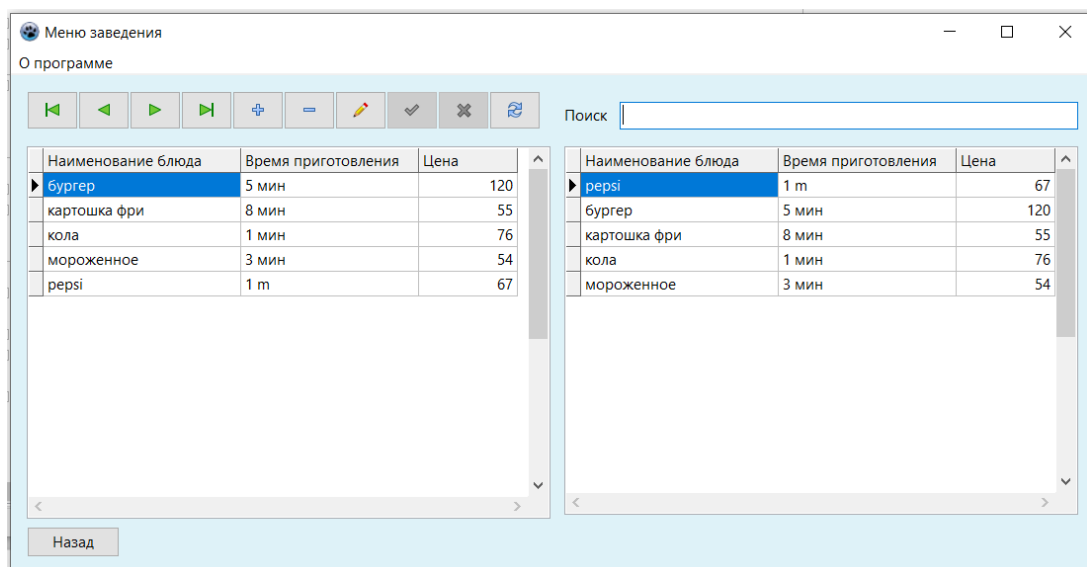
Тестирование удаления записи дало следующие результаты. Таблица до удаления показана на рисунке 11



Наименование блюда	Время приготовления	Цена
паста	34 мин	530
бургер	5 мин	120
картошка фри	8 мин	55
кола	1 мин	76
мороженное	3 мин	54
перси	1 м	67

Рисунок 11. Таблица до удаления записи

Таблица после удаления показана на Рисунке 12.



Наименование блюда	Время приготовления	Цена
бургер	5 мин	120
картошка фри	8 мин	55
кола	1 мин	76
мороженное	3 мин	54
перси	1 м	67

Рисунок 12. Таблица после удаления записи

Также проводилось тестирование работы редактирование записи, что показало нам следующее:

Таблица до редактирования выбранной записи показана на рисунке 13.

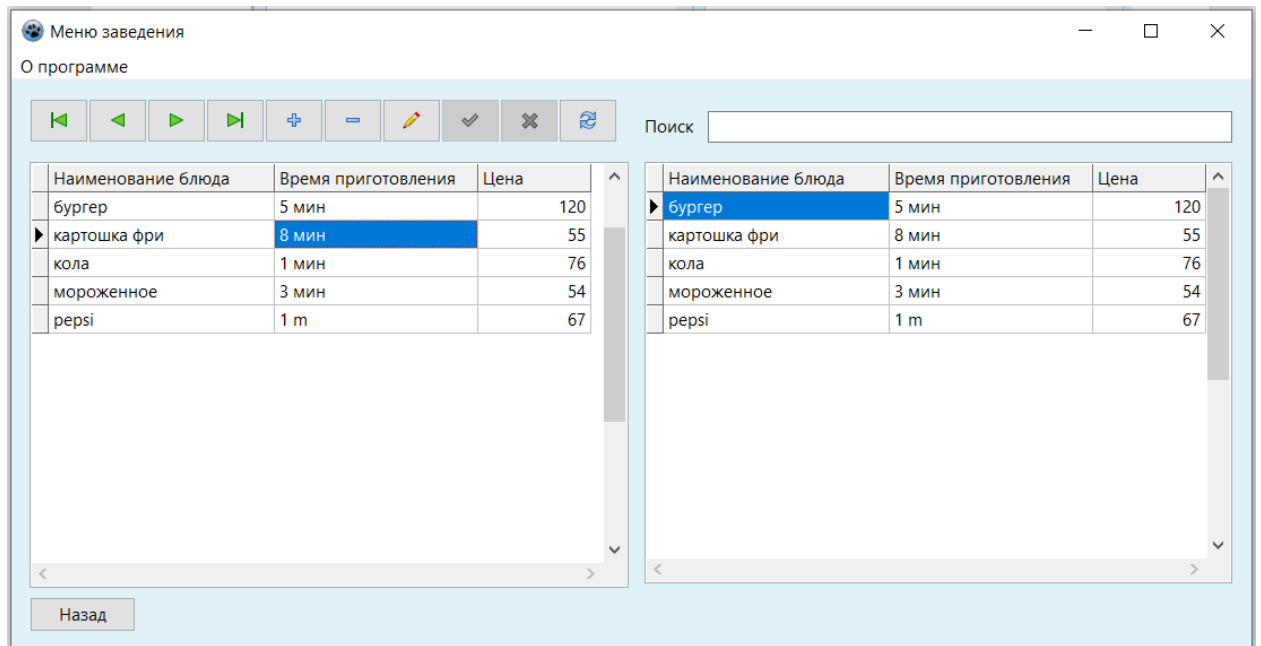


Рисунок 13. Таблица до редактирования

Таблица после редактирования показана на рисунке 14

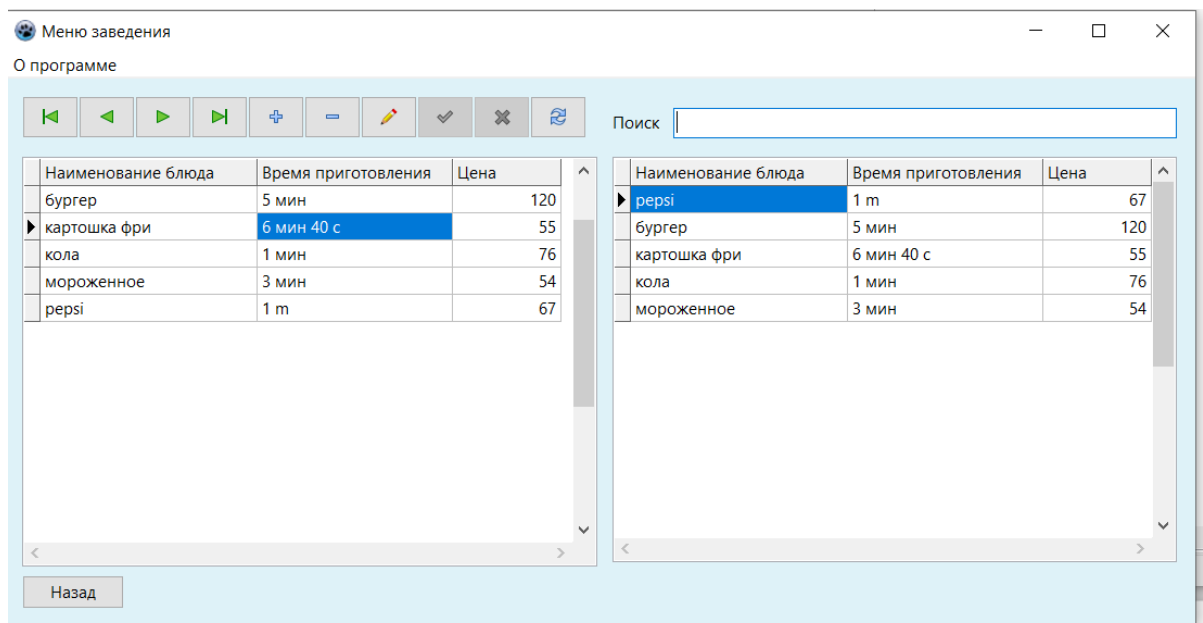


Рисунок 14. Таблица после редактирования

Тестирование поиска осуществлялось на английском, русском языках, а также на числах. При тестировании поиска обнаружилась недоработка, которую планируется устранить в следующей версии приложения. При поиске на русском языке выходит системная ошибка. (Рисунок 18)

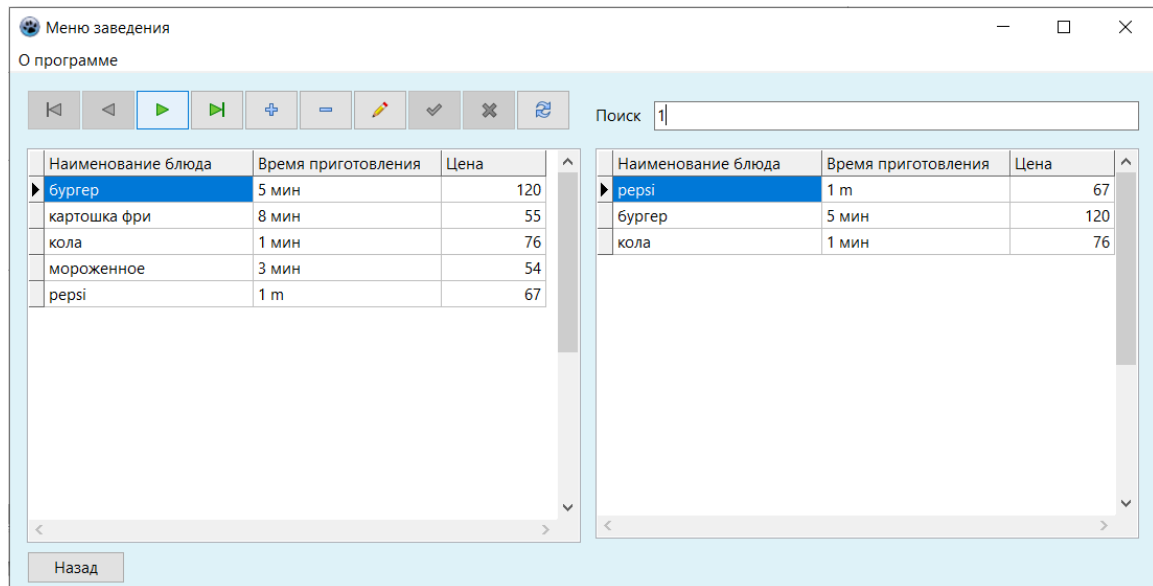


Рисунок 15. Поиск 1

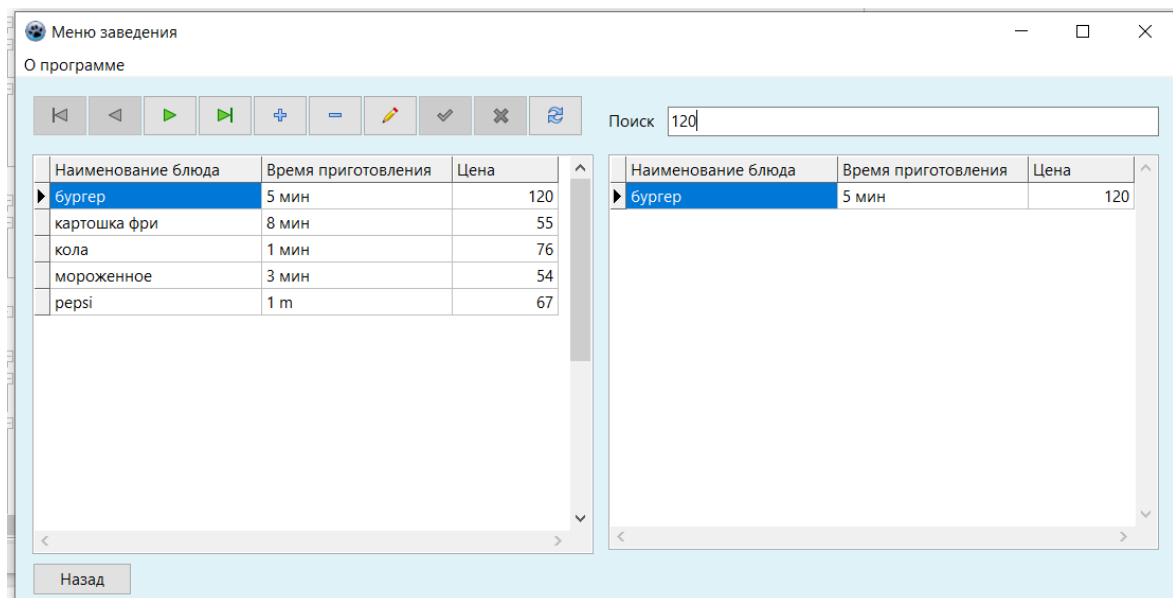


Рисунок 16. Поиск числовой информации

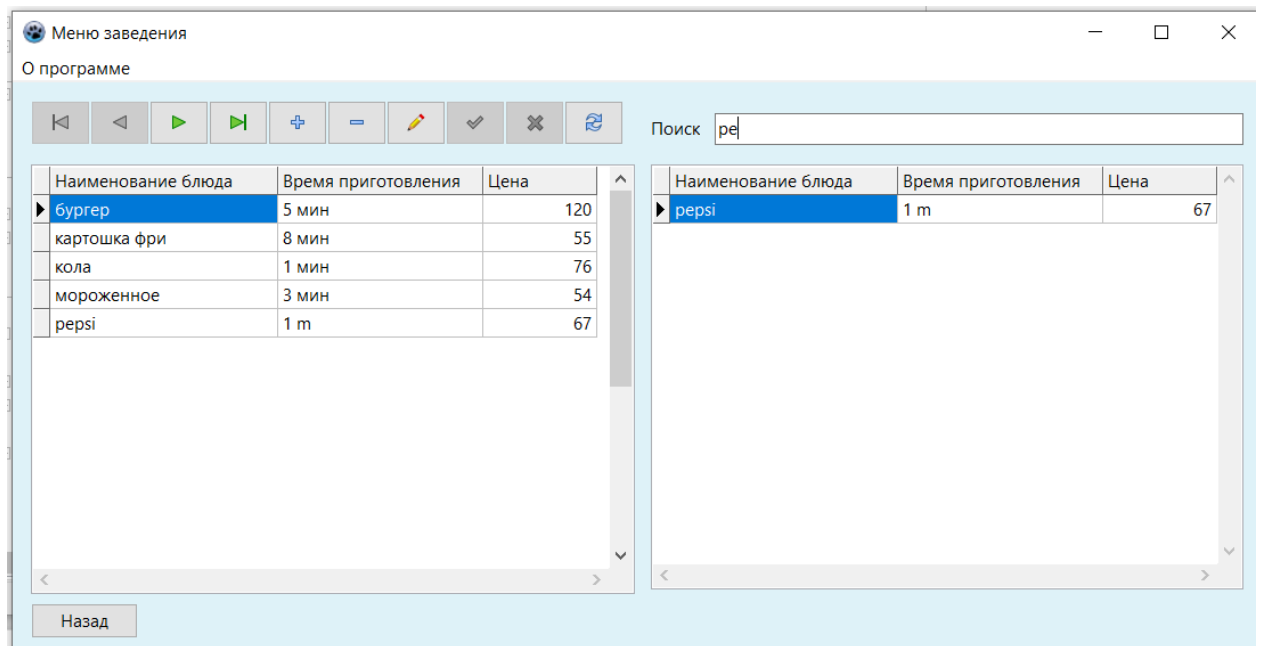


Рисунок 17. Поиск на английском языке

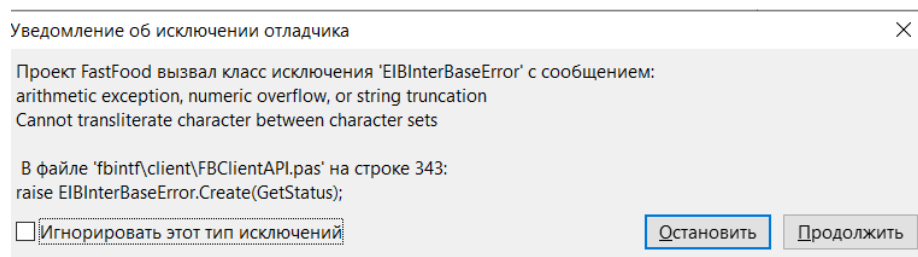


Рисунок 18. Системная ошибка

2.9. Описание применения средств отладки

При компиляции проекта были обнаружены ошибки. Сообщение отладчика гласит, что на строке 70 модуля finance обнаружено неизвестное имя.

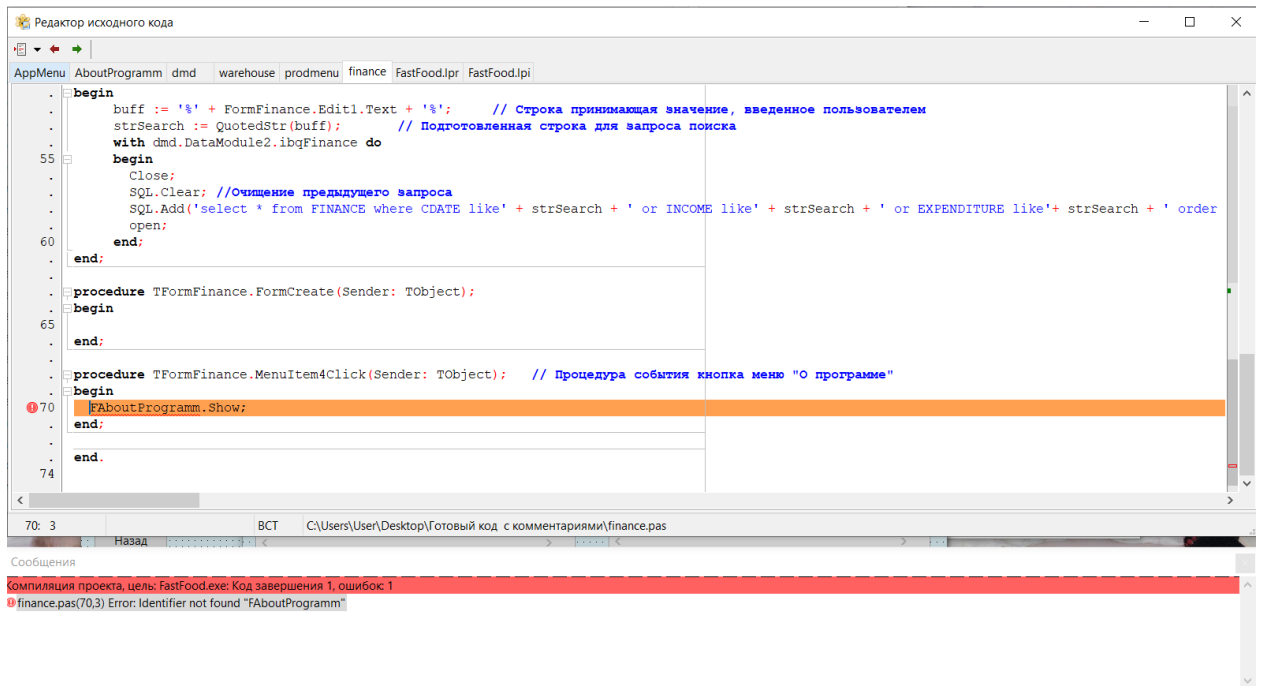


Рисунок 19. Ошибка

Решением данной проблемы стало подключение модуля, где происходит объявление данного имени.

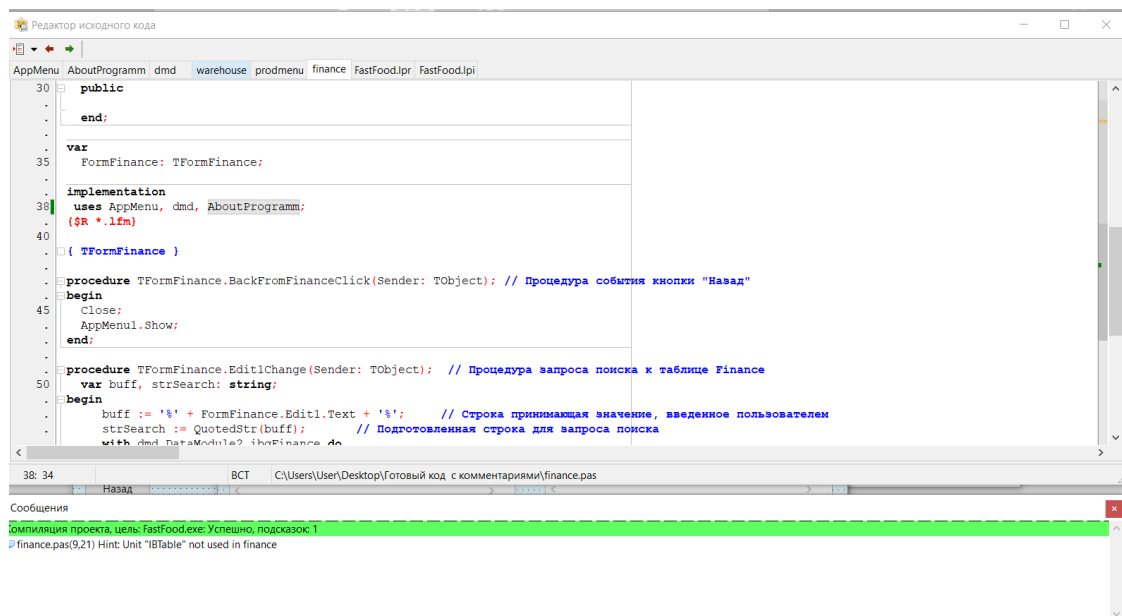
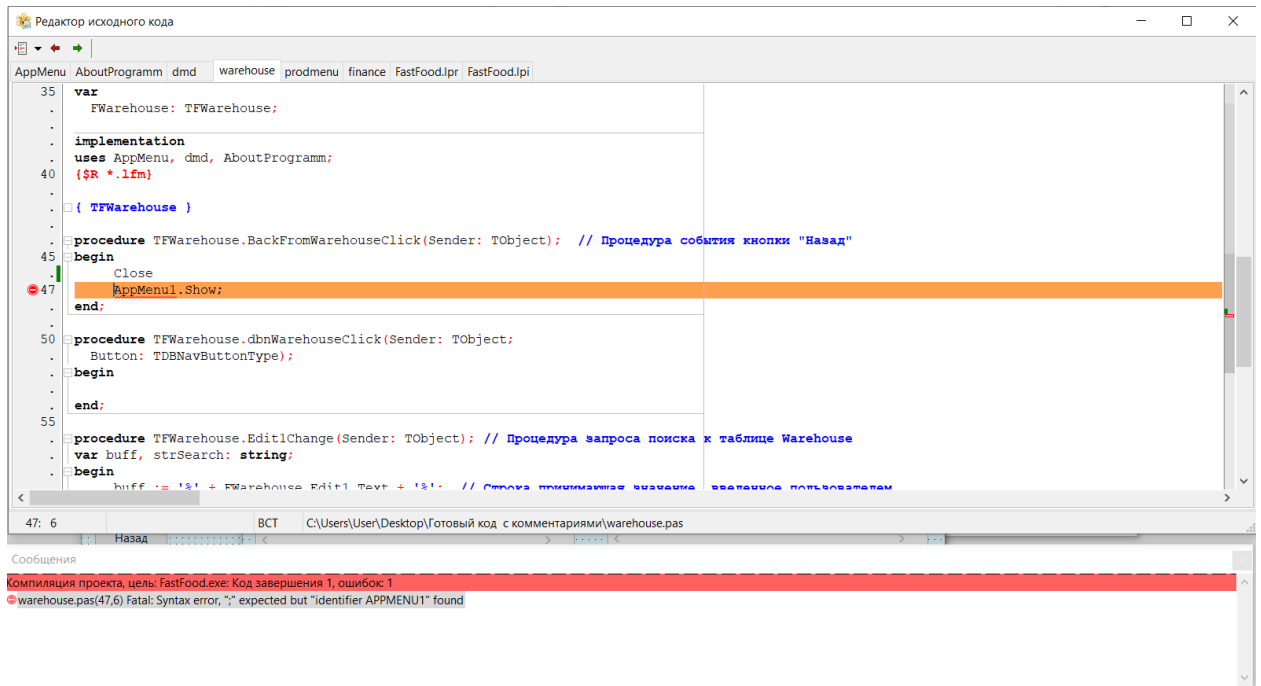
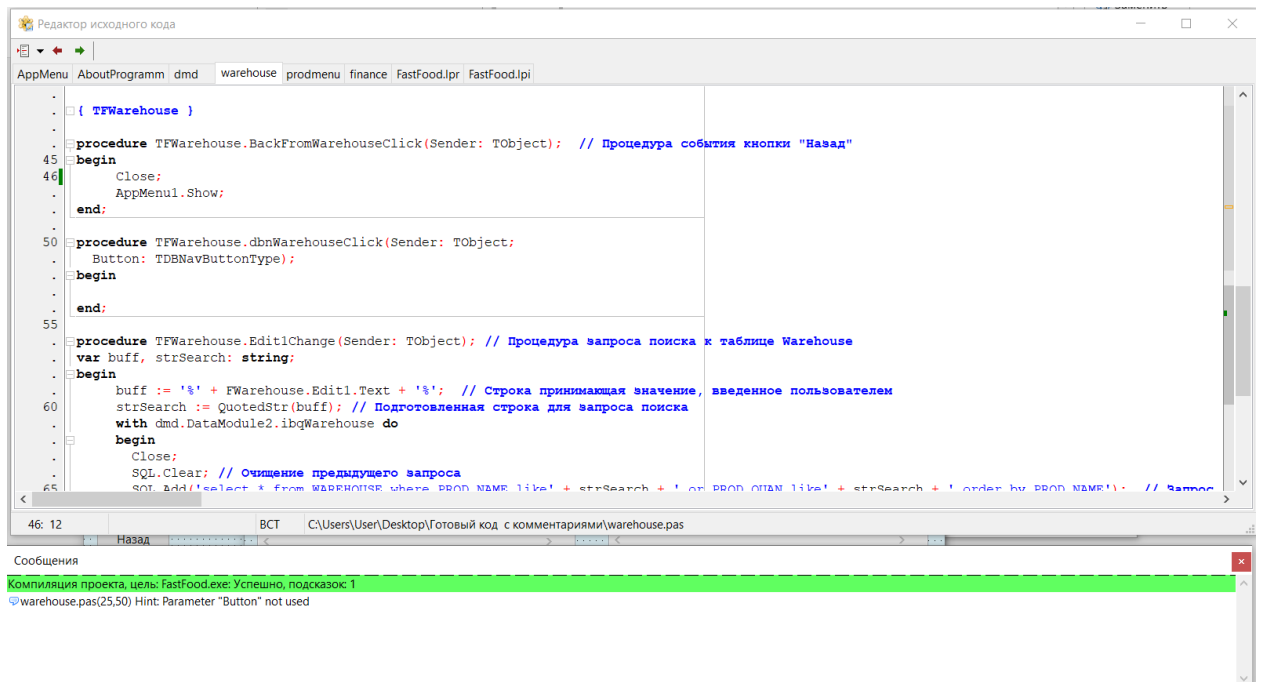


Рисунок 20. Успешная компиляция

В модуле warehouse была обнаружена синтаксическая ошибка.



Сообщения компилятора после компиляции проекта представлены на Рисунке 22



2.10. Анализ оптимальности использования памяти и быстродействия

Удаление мертвого кода является одним из методов оптимизации. Для быстродействия программы было решено воспользоваться данным методом.

Были удалены следующие части кода:

Листинг 2.10.1. Мертвый код в модуле warehouse

```
procedure TFWarehouse.FormCreate(Sender: TObject);
begin

end;
```

Листинг 2.10.1. Мертвый код в модуле prodmenu

```
procedure TFProdMenu.FormCreate(Sender: TObject);
begin

end;
```

Листинг 2.10.1. Мертвый код в модуле AppMenu

```
procedure TAppMenu1.Button4Click(Sender: TObject);
begin

end;

procedure TAppMenu1.Label1Click(Sender: TObject);
begin

end;
```

Размер файла FastFood.exe достаточно большой. Было решено оптимизировать компиляцию, тем самым уменьшить размер файла. Размер файла до оптимизации показан ниже


 FastFood.exe	17.06.2020 11:51	Приложение	30 350 КБ
--	------------------	------------	-----------

Рисунок 23. Размер файла до оптимизации

Для этого было необходимо отключить отладочную информацию и включить «вырезать символы из исполняемого файла».

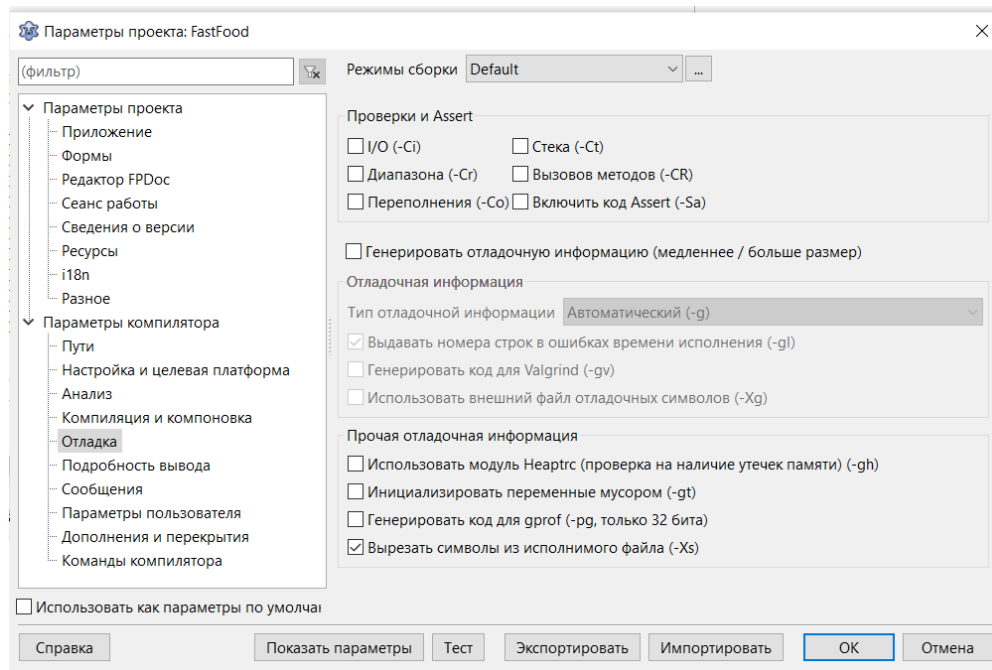


Рисунок 24. Изменение параметров отладки

Также нужно включить умную компоновку в «Стиль модуля» и в «Компоновка»

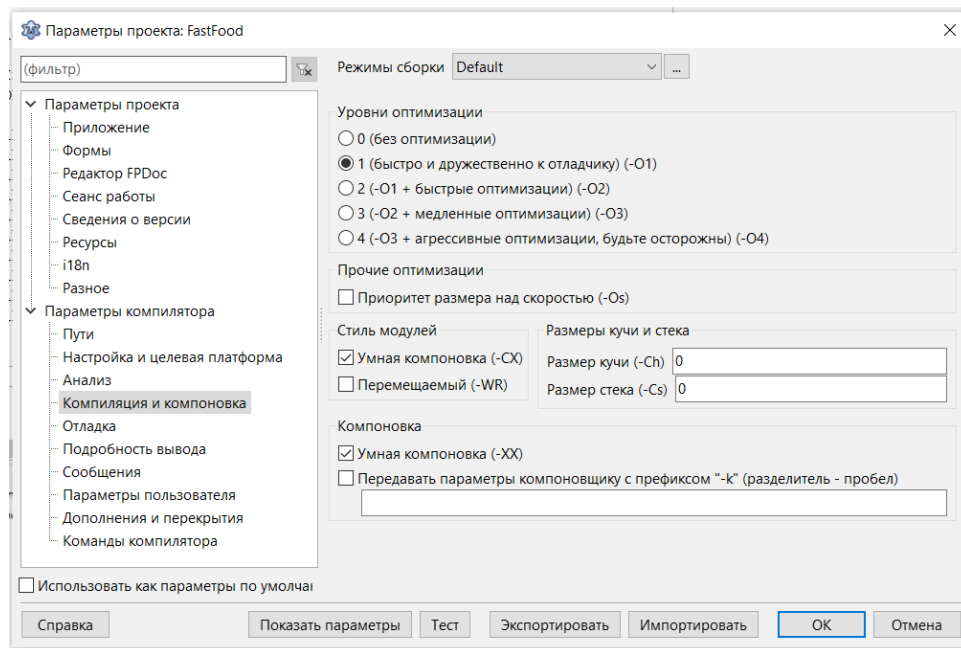


Рисунок 25. Параметры компиляции и компоновки

Для уменьшения памяти можно убрать стандартный значок

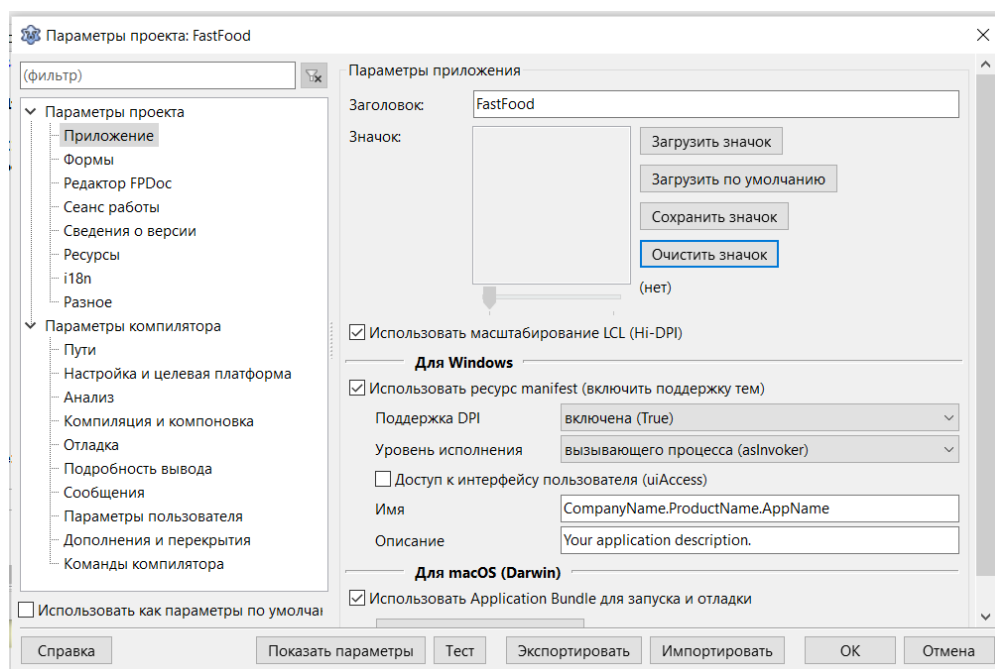


Рисунок 26. Очистение стандартного значка

После проведенной оптимизации компиляции можно заметить существенную разницу в размере файла. До оптимизации размер exe-файла составлял 30350 КБ, а после 4187 КБ

FastFood.exe	20.06.2020 13:20	Приложение	4 187 КБ
--------------	------------------	------------	----------

Рисунок 27. Размер файла после оптимизации

Глава 3. Эксплуатационная часть

3.1. Руководство оператора

3.1.1. Назначение программы

Разработанное приложение позволяет осуществлять различный учет данных на предприятии быстрого питания. При этом пользователю не обязательно обладать умением работы с базами данных. Программа работает с таблицами БД FireBird – записывает все изменения, внесенные пользователем в таблицу.

3.1.2. Условия выполнения программы

Минимальные требования:

Операционная система: Windows: 7 x 32bit;

3.1.3. Выполнение программы

После запуска приложения открывается окно выбора (Рисунок 28), содержащее три кнопки «Меню заведения», «Склад», «Финансы», а также в верхней панели «О программе»

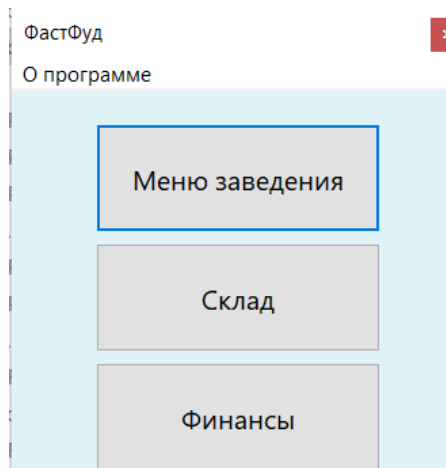


Рисунок 28. Окно выбора «ФастФуд»

При нажатии на «О программе» открывается окно, содержащее информацию о разработчике, а также о теме курсового проекта, которой посвящено данное приложение. (Рисунок 29)

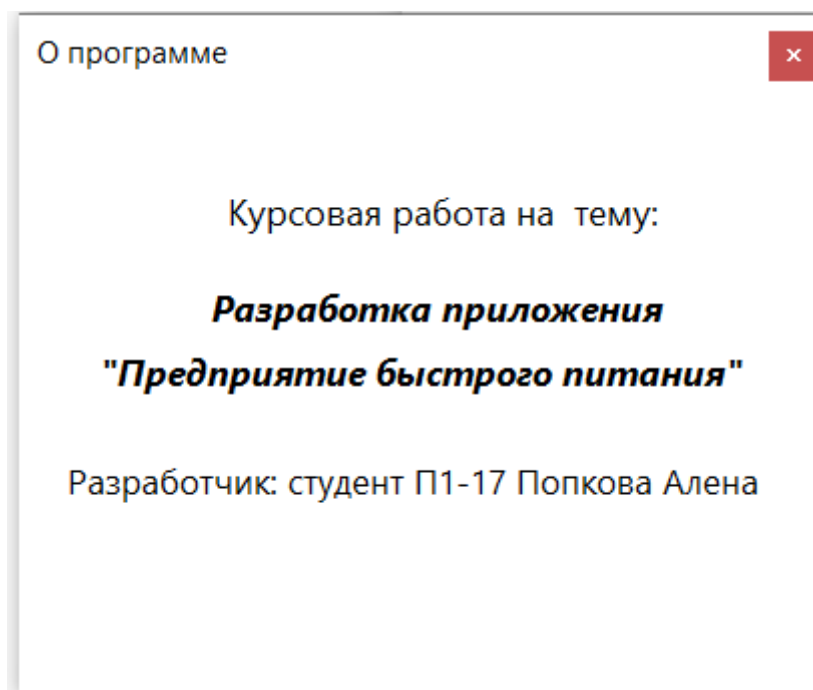


Рисунок 29. Окно «О программе»

При нажатии на одну из всех предложенных кнопок, открывается соответствующее окно.

Кнопка «Меню заведения» откроет окно «Меню заведения» с соответствующей таблицей.

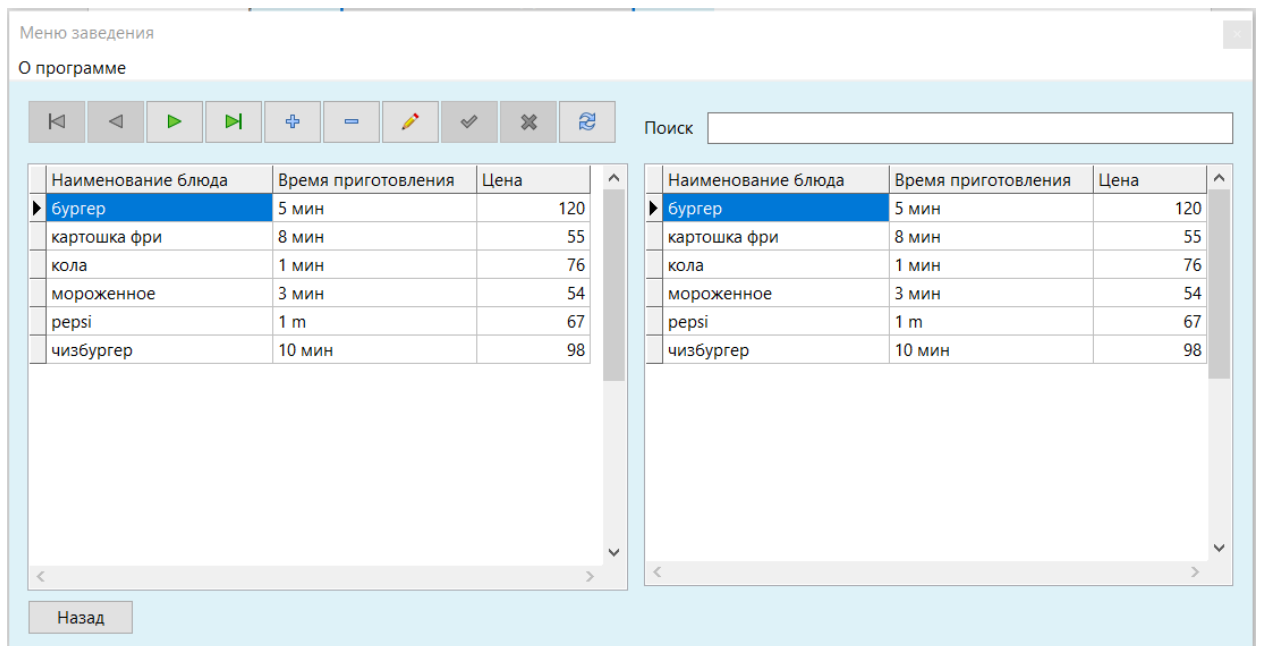


Рисунок 30. Окно «Меню заведения»

Кнопка «Склад» откроет окно «Склад» с соответствующей таблицей. (Рисунок 31)

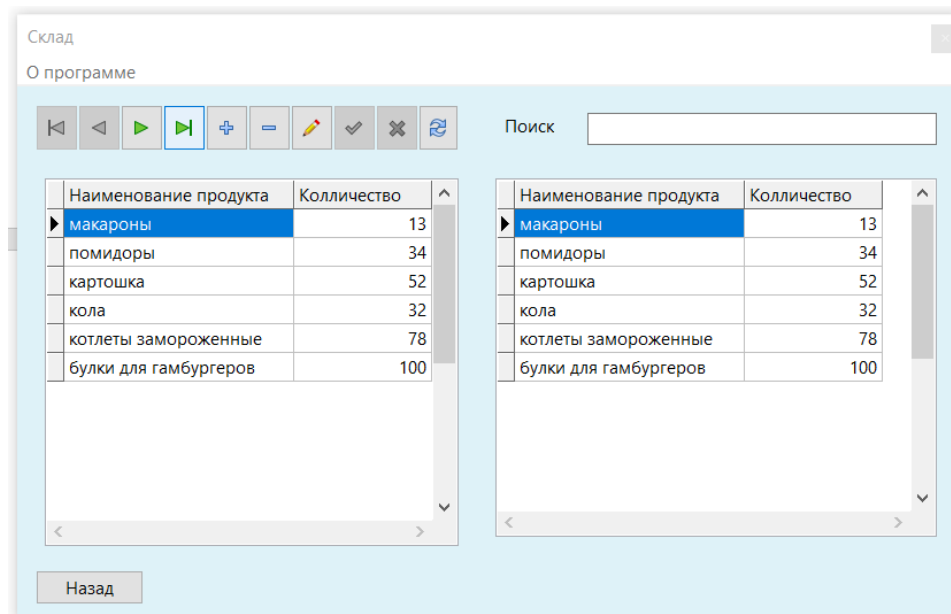


Рисунок 31. Окно «Склад»

Кнопка «Финансы» откроет окно «Финансы» с соответствующей таблицей. (Рисунок 32)

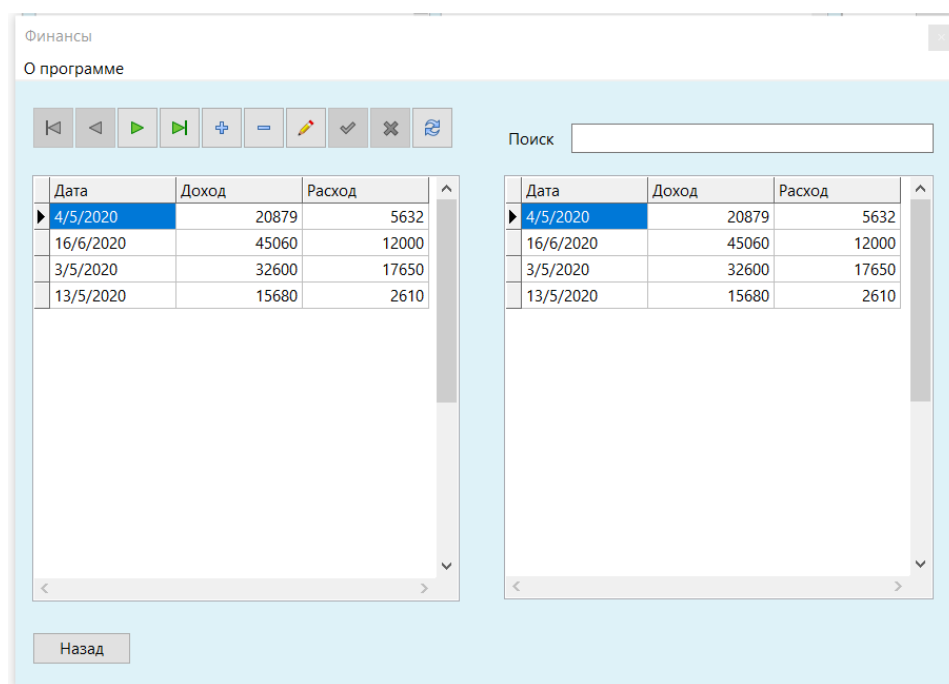


Рисунок 32. Окно «Финансы»

Каждое из окон имеет примерно одинаковый интерфейс. Описание функционала окна представлено на рисунке ниже (Рисунок 33). При нажатии на кнопку «Назад» открытое окно закрывается и пользователю открывается окно выбора «Фастфуд» (Рисунок 28)

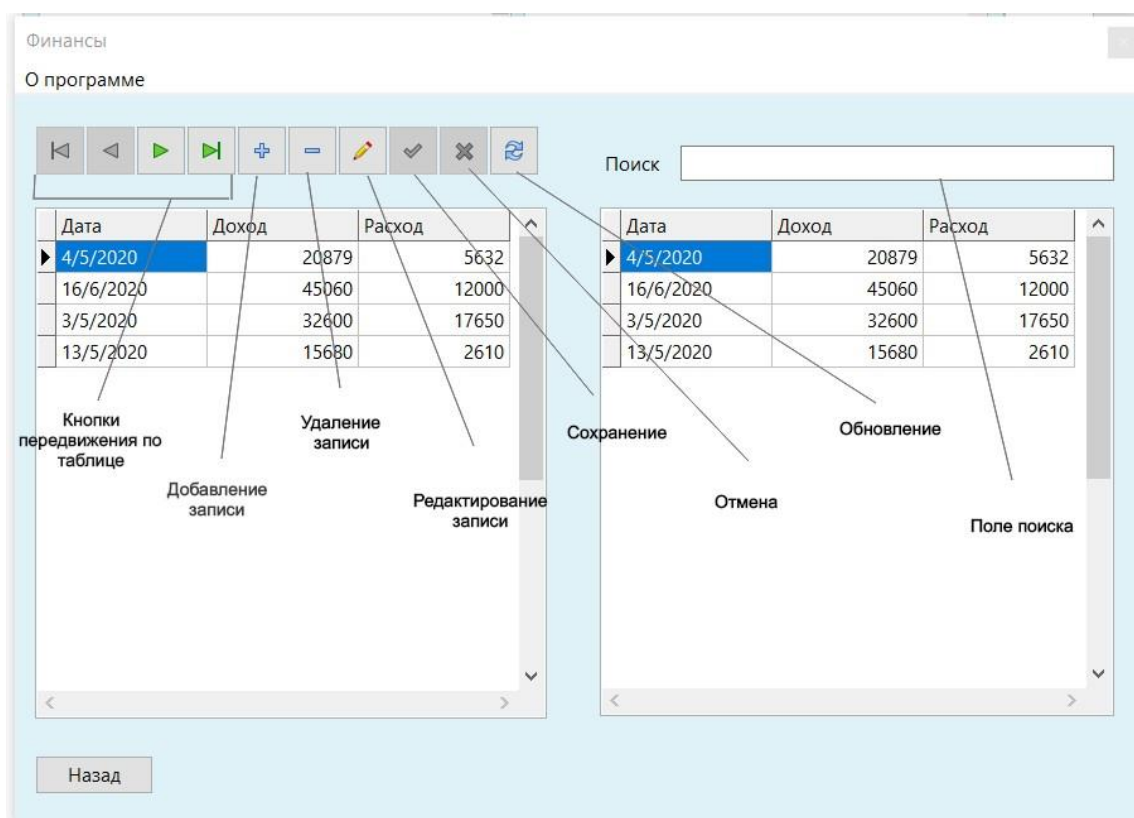


Рисунок 33. Описание функционала окон

3.1.4. Сообщение оператору

При незаполнении полей «Наименование продукта» в таблицу «Склад», «Наименование блюда» в таблице «Меню заведения» и «дата» в таблице «Финансы» программа выдает сообщение об ошибке. (Рисунок 34)

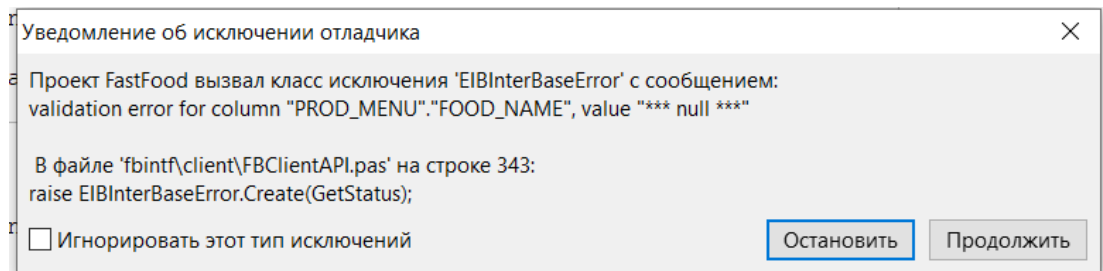


Рисунок 34. Системная ошибка

Заключение

В результате выполнения курсового проекта было разработано приложение «Предприятие быстрого питания» для учета различной информации, а именно для бухгалтерского и складского учета.

Данное приложение может использоваться в учебных целях.

В ходе работы были проанализированы существующие разработки посвященные данному направлению и изучены следующие темы: «Работа с базами данных FireBird» и «Разработка приложения с базами данных».

В дальнейшем планируется разработать окно заказа, автоматическое списание продуктов со склада, а также обработчик ошибок при заполнении таблицы.

Список литературы и интернет-источников

1. Официальный сайт 1С:Предприятие 8. Фастфуд. Фронт-офис:
<https://solutions.1c.ru/catalog/fastfood/features>
2. Официальный сайт Lazarus
<https://www.lazarus-ide.org/>
3. Официальный сайт FireBird:
<https://www.firebirdsql.org/en/server-packages/>
4. Статья «Быстрое питание»:
https://ru.wikipedia.org/wiki/%D0%91%D1%8B%D1%81%D1%82%D1%80%D0%BE%D0%B5_%D0%BF%D0%B8%D1%82%D0%B0%D0%BD%D0%B8%D0%B5
5. Статья «Рестораны быстрого питания»:
https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D1%81%D1%82%D0%BE%D1%80%D0%B0%D0%BD%D1%8B_%D0%B1%D1%8B%D1%81%D1%82%D1%80%D0%BE%D0%B3%D0%BE_%D0%BF%D0%B8%D1%82%D0%B0%D0%BD%D0%B8%D1%8F

Приложение 1. Модуль AboutProgramm

```
unit AboutProgramm;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls;

type

  { TFAboutProgramm }

  TFAboutProgramm = class(TForm)    //Класс формы "О программе"
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    procedure FormCreate(Sender: TObject);
  private
  public
  end;

var
  FAboutProgramm: TFAboutProgramm;

implementation

{$R *.lfm}

{ TFAboutProgramm }

procedure TFAboutProgramm.FormCreate(Sender: TObject);
begin

end;

end.
```

Приложение 2. Модуль prodmenu

```

unit prodmenu;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Menus,
  DBGrids, DBCtrls;

type

  { TFProdMenu }

  TFProdMenu = class(TForm)           // Класс формы "Меню
    предприятия"
    ButMenuBack: TButton;              // кнопка "Назад"
    DBGrid1: TDBGrid;                  // компонент для отображения
    данных таблицы БД
    DBGrid2: TDBGrid;                  // компонент для отображения
    данных таблицы БД
    dbnProdMenu: TDBNavigator;          // компонент для работы с
    данными таблицы БД
    Edit1: TEdit;                      // поле ввода
    Label1: TLabel;                    // надпись
    MainMenu1: TMainMenu;              // невизуальный компонент меню
    MenuItem4: TMenuItem;              // пункт меню
    procedure ButMenuBackClick(Sender: TObject); // процедура
    события кнопки "Назад"
    procedure Edit1Change(Sender: TObject); // обработчик события
    Edit
    procedure MenuItem4Click(Sender: TObject);
  private
  public

  end;

var
  FProdMenu: TFProdMenu;

implementation
uses AppMenu, dmd, AboutProgramm;
{$R *.lfm}

{ TFProdMenu }

```

```

procedure TFProdMenu.ButMenuBackClick(Sender: TObject); // Процедура
события кнопки "Назад"
begin
    Close;
    AppMenu1.Show;
end;

procedure TFProdMenu.Edit1Change(Sender: TObject); // Процедура
запроса поиска к таблице ProdMenu
var buff, strSearch: string;
begin
    buff := '%' + FProdMenu.Edit1.Text + '%'; // Строка принимающая
значение, введенное пользователем
    strSearch := QuotedStr(buff); // Подготовленная строка для
запроса поиска
    with dmd.DataModule2.ibqProdmenu do
    begin
        Close;
        SQL.Clear; //Очищение предыдущего запроса
        SQL.Add('select * from PROD_MENU where FOOD_NAME like' +
strSearch + ' or COOKING_TIME like' + strSearch + ' or PRICE like' +
strSearch + ' order by FOOD_NAME'); // Запрос поиска
        open;
    end;
end;

procedure TFProdMenu.MenuItem4Click(Sender: TObject); // Процедура
события кнопка меню "О программе"
begin
    FAboutProgramm.Show;
end;

end.

```

Приложение 3. Модуль finance

```

unit finance;

{$mode objfpc}{$H+}

interface

uses
  Classes, SysUtils, db, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Menus,
  DBGrids, DBCtrls, IBTable;

type

  { TFormFinance }

  TFormFinance = class(TForm)           // Клас формы "Финансы"
    BackFromFinance: TButton;           // кнопка "Назад"
    DBGrid1: TDBGrid;                   // компонент для отображения
    данных таблицы БД
    DBGrid2: TDBGrid;                   // компонент для отображения
    данных таблицы БД
    dbnFinance: TDBNavigator;           // компонент для работы с
    данными таблицы БД
    Edit1: TEdit;                       // поле ввода
    Label1: TLabel;                     // надпись
    MainMenu1: TMainMenu;               // невидимый компонент меню
    MenuItem4: TMenuItem;               // пункт меню
    procedure BackFromFinanceClick(Sender: TObject); // процедура
    события кнопки "Назад"
    procedure Edit1Change(Sender: TObject); // обработчик
    события Edit
    procedure MenuItem4Click(Sender: TObject);
  private
  public

  end;

var
  FormFinance: TFormFinance;

implementation
  uses AppMenu, dmd, AboutProgramm;
  {$R *.lfm}

  { TFormFinance }

  procedure TFormFinance.BackFromFinanceClick(Sender: TObject); //
  Процедура события кнопки "Назад"

```



```

begin
    Close;
    AppMenu1.Show;
end;

procedure TFormFinance.Edit1Change(Sender: TObject); // Процедура
запроса поиска к таблице Finance
var buff, strSearch: string;
begin
    buff := '%' + FormFinance.Edit1.Text + '%'; // Строка
принимаящая значение, введенное пользователем
    strSearch := QuotedStr(buff); // Подготовленная строка для
запроса поиска
    with dmd.DataModule2.ibqFinance do
    begin
        Close;
        SQL.Clear; //Очищение предыдущего запроса
        SQL.Add('select * from FINANCE where CDATE like' + strSearch +
' or INCOME like' + strSearch + ' or EXPENDITURE like'+ strSearch + '
order by CDATE'); // Запрос поиска
        open;
    end;
end;

procedure TFormFinance.MenuItem4Click(Sender: TObject); // Процедура
события кнопка меню "О программе"
begin
    FAboutProgramm.Show;
end;

end.

```

Приложение 4. Модуль данных dmd

```

unit dmd;

{$mode objfpc}{$H+}

interface

uses
    Classes, SysUtils, IBDatabase, IBTable, IBCustomDataSet, IBQuery,
    db, INIFiles;

type

    { TDataModule2 }

    TDataModule2 = class(TDataModule)      // Класс "Модуль данных"
    dtsWarehouseQ: TDataSource;
    dtsFinanceQ: TDataSource;
    dtsProdmenuQ: TDataSource;
    dtsWarehouse: TDataSource;
    dtsFinance: TDataSource;
    dtsProdMenu: TDataSource;
    ibdb: TIBDatabase;
    ibFinanceCDATE: TIBStringField;
    ibFinanceEXPENDITURE: TIBIntegerField;
    ibFinanceINCOME: TIBIntegerField;
    ibProdmenuCOOKING_TIME: TIBStringField;
    ibProdmenuFOOD_NAME: TIBStringField;
    ibProdmenuPRICE: TIBIntegerField;
    ibqFinanceCDATE: TIBStringField;
    ibqFinanceEXPENDITURE: TIBIntegerField;
    ibqFinanceINCOME: TIBIntegerField;
    ibqProdmenuCOOKING_TIME: TIBStringField;
    ibqProdmenuFOOD_NAME: TIBStringField;
    ibqProdmenuPRICE: TIBIntegerField;
    ibqWarehouse: TIBQuery;
    ibqFinance: TIBQuery;
    ibqProdmenu: TIBQuery;
    ibqWarehousePROD_NAME: TIBStringField;
    ibqWarehousePROD_QUAN: TIBIntegerField;
    ibWarehouse: TIBTable;
    ibFinance: TIBTable;
    ibProdmenu: TIBTable;
    ibtr: TIBTransaction;
    ibWarehousePROD_NAME: TIBStringField;
    ibWarehousePROD_QUAN: TIBIntegerField;
    procedure DataModuleCreate(Sender: TObject);
    private

    public

```

```

end;

var
  DataModule2: TDataModule2;
  IniF : TINIFile;
  DBname : string;

implementation

{$R *.lfm}

{ TDataModule2 }

procedure TDataModule2.DataModuleCreate(Sender: TObject);
// Запуск транзакции и открытие таблиц
begin
  IF(FileExists('DataBase.ini'))then // Проверка ini-файла
  begin
    IniF := TINIFile.Create('DataBase.ini');
    DBname := IniF.ReadString('db', 'dbname', '');
  // Считывание пути файла

    ibdb.DatabaseName := DBname;
    ibdb.Connected := True;
    ibdb.AllowStreamedConnected := True;
    ibtr.StartTransaction; // Запуск транзакции
    ibWarehouse.Open;      // Открытие таблицы Warehouse
    ibqWarehouse.Open;     // Открытие таблицы Warehouse для
запроса поиска
    ibProdmenu.Open;       // Открытие таблицы Prodmenu
    ibqProdmenu.Open;      // Открытие таблицы Prodmenu для
запроса поиска
    ibFinance.Open;        // Открытие таблицы Finance
    ibqFinance.Open;       // Открытие таблицы Finance для
запроса поиска
  end;
end;

end.

```