

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

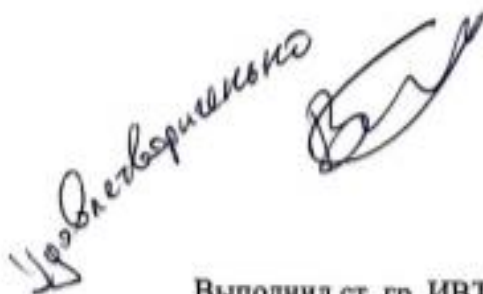
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический факультет Кафедра: Информатики и
вычислительной техники

Курсовая работа

по Программированию

На тему: База данных для регистрации посетителей ресторана



Выполнил ст. гр. ИВТ-20-2 Замешаев А.И.

Проверил Доцент кафедры ИВТ и ПМ
Соловьёв В.А.

Чита

2021

Задание для Курсовой работы

Создать программу, в которой создается база данных для записи посетителей ресторана, сведения включают: номер заказа, Ф.И.О. обслужившего, перечень заказанных блюд, сумму оплаты. В конце дня подводится итог.

Программа должна предоставить возможность просматривать, добавлять, удалять, копировать, хранить данные.

В программе использовать модули, функции, процедуры, записи, списки и файлы

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования

«Забайкальский государственный университет»

(ФГБОУ ВО «ЗабГУ»)

Факультет: Энергетический факультет

Кафедра: Информатики, вычислительной техники и прикладной
математики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе (проекту)

по Программированию на тему: База данных для регистрации
посетителей ресторана

Выполнил студент группы ИВТ-20-2 Замешаев А.И

Руководитель работы: Ветров С.В.

СОДЕРЖАНИЕ

Введение.....	3
Цели курсовой работы:.....	3
Глава 1. Типы данных и операции, реализуемые в КР	4
1.1 Типы данных реализуемые в КР	4
1.2 Операции реализуемые в КР	4
Глава 2.Интерфейс приложения	6
2.1 Описание компонентов использованных в курсовой работ	6
2.2 Реализация обработчиков событий.....	8
Глава 3. Тестирование приложения и проверка результатов	12
Заключение	16
Литература	17
Приложения	18

Введение

Цель работы – разработка приложения, с минимальным интерфейсом, которое будет работать с базой данных для записи посетителей ресторана. Также целью будет являться: Применение типов данных: stringred, integer, real, record, изученных во время обучения, использование процедур и функций, модулей, файлов, списков и записей.

Цели курсовой работы:

- применение типов данных, изученных в течение двух семестров;
- использование процедур, функций, модулей, файлов и списков;
- создание минимального интерфейса Delphi;
- создание простейшей базы данных.

Глава 1. Типы данных и операции, реализуемые в КР

1.1 Типы данных реализуемые в КР

В данной работе используется список из записей, который можно хранить в файле, потому нам необходим тип записи, содержащий в себе основные элементы базы данных, такие как, номер заказа, Ф.И.О. обслуживающего, заказанные блюда, стоимость заказа и итог дня. Помимо этого, нам необходим тип файла, для хранения базы данных, а так же, ещё одна запись, в которой содержится указатель на предыдущий и следующий узлы и на ранее названную запись.

Секция Туре данной работы:

```
type
  Zak = record           //Тип заказа
    Number: real;        //номер стола
    FIO : string[64];    //Фио официанта
    Zakaz: string [255]; //перечень блюд
    SumZak: real;        //сумма заказа
    ItogDay: real;       //итог дня
  end;
  PUzel = ^Zp;           //Тип указателя на узел
  Zp = record            //Тип узла списка
    x: Zak;              //Информация хранящаяся в узлах списка
    next: PUzel;         //Указатель на следующий узел
    pred: PUzel;         //Указатель на предыдущий узел
  end;
  FZap = file of Zak;    //Файловый тип для хранения данных о заказе
```

1.2 Операции реализуемые в КР

В данной работе предоставляется возможность просматривать, добавлять , удалять, хранить и обрабатывать данные. Для этого в модуле программы описаны следующие процедуры:

```
{Операции для работы с двусвязанным списком }
procedure AddFirst(var f: PUzel; a: PUzel);{Вставить узел а первым в список}
procedure AddAfter(var old:PUzel; a: PUzel);{Вставить узел а после old}
```

```

{Построить список; f -указатель на голову списка}
procedure BuildSpisok(var f: PUzel; var ftxt: TextFile);

{Вывод списка в текстовый файл}
procedure WriteSpText(var f: PUzel;var ftxt:Text);

{Выделить из списка первый узел и вернуть его пользователю}
procedure DelFirstElement(var f,a: PUzel);

{Выделить из списка узел,следующий за узлом old и вернуть его пользователю}
procedure DelElement(var old,a: PUzel);
procedure DelSpisok(var f: PUzel); //Удалить список

```

Рассмотрим примеры реализаций из представленных операций:

Содержимое двусвязанного списка

1 ФИО официанта:Шарин Владимир Александрович Заказанные блюда:чай, макароны, котлета по киевски, салат греческий Сумма заказа:280 Итог дня:280

Содержимое двусвязанного списка

2 ФИО официанта:Грушин Александр Витальевич Заказанные блюда:компот, суп гороховый, рис с курицей Сумма заказа:250 Итог дня:530

Содержимое двусвязанного списка

3 ФИО официанта:Шишкин Петр Петрович Заказанные блюда:чай с лимоном, спагетти сыром, котлета рыбная Сумма заказа:240 Итог дня:770

Код подключенного модуля построения списка

```

{Построить список; f -указатель на голову списка}
procedure BuildSpisok(var f: PUzel; var ftp: FZap);
var
  a,d :PUzel;
  ch: char;
begin
  repeat
    new(a);

```

```

with a^.x do
begin
    Number:= StrToInt(TextBox ('Введите номер заказа', ' ', ' '));
//Ввод номера стола (с выводом на экран окна ввода)
    FIO := TextBox('Введите ФИО официанта', ' ', ' ');
//Ввод ФИО официанта (с выводом на экран окна ввода)
    Zakaz:= TextBox('Введите список заказанных блюд', ' ', ' ');
//Ввод Заказанных блюд (с выводом на экран окна ввода)
    SumZak:= StrToFloat(TextBox('Введите сумму заказа', ' ', ' '));
//Ввод суммы заказа (с выводом на экран окна ввода)
end;

a^.next := nil;
a^.pred := nil;

if (f = nil) then
begin
    AddFirst(f,a);
    d:= f;
end
else
begin
    AddAfter(d,a);
    d := a;
end;
ch:= TextBox('Для завершения ввода нажмите 0', ' ', ' ')[1];
until (ch = '0') or (ch = '0');
end;

```

Глава 2.Интерфейс приложения

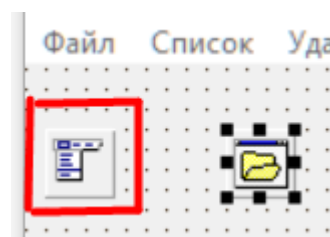
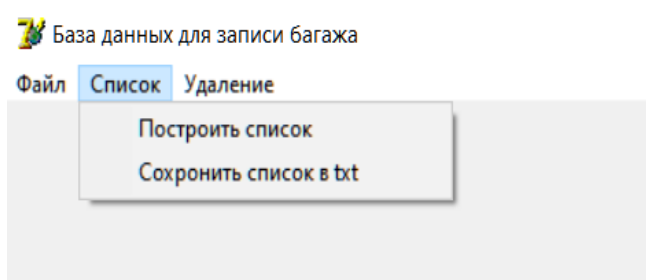
Интерфейс данной курсовой работы должен иметь хороший функционал и при этом быть максимально простым для пользователя. С целью соответствия этим критериям в данной работе были использованы такие компоненты как MainMenu, OpenFileDialog.

2.1 Описание компонентов использованных в курсовой работе

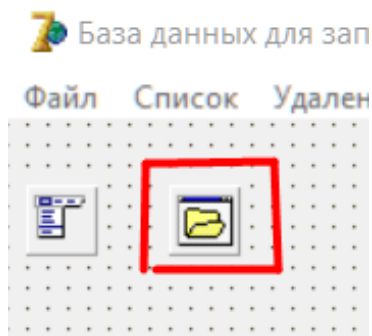
Одним из основных компонентов данной курсовой работы является MainMenu, добавляющий в приложение главное меню, стандартный элемент

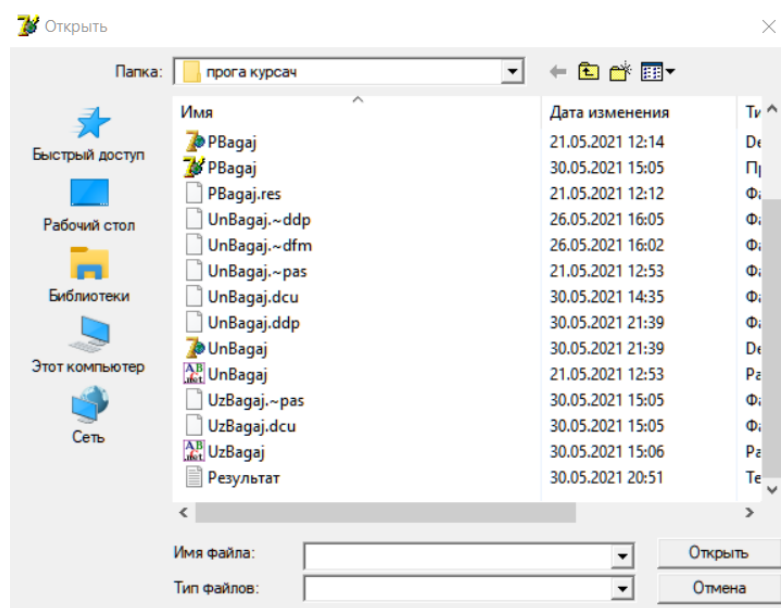
для всех приложений Windows. За счёт него форма не нагружается огромным количеством кнопок.

Меню представляет собой список из пунктов, объединенных по функциональному признаку. Каждый пункт меню обозначает либо вложенное подменю, либо конкретную команду. Поскольку MainMenu является визуальным компонентом то его можно располагать в любом месте на форме, поскольку пользователь увидит лишь конечный результат работы данного компонента.

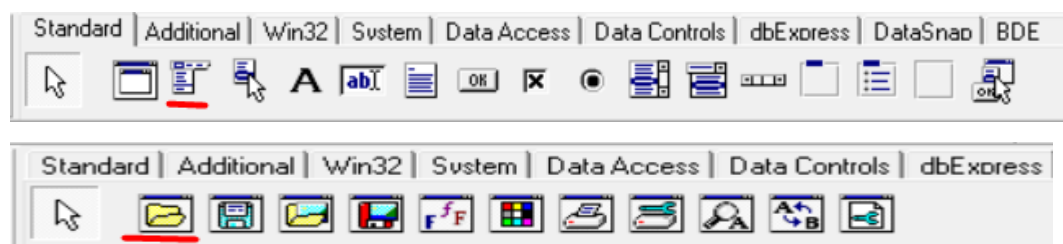


Далее рассмотрим компонент OpenDialog. Это не визуальный компонент, который используется для поддержки операций открытия файлов. Он способен работать с любыми видами файлов. При обращении к этому компоненту вызывается стандартное диалоговое окно открытия файла.





Все два вышеназванных компонента находятся на панели инструментов во вкладке Standart и Dialogs:



2.2 Реализация обработчиков событий

Здесь будут представлены коды обработчиков событий, используемых в данной работе.

Процедура открытия текстового файла

```
procedure TForm1.OpenTxtFile1Click(Sender: TObject);
```

```
var
```

```
  s: string;
```

```
begin
```

```
  if not OpenDialog1.Execute then exit;
```

```
s := OpenFileDialog.FileName;  
AssignFile(ftxt,s);  
Append(ftxt)  
end;
```

Процедура закрытия текстового файла

```
procedure TForm1.CloseTxtFile1Click(Sender: TObject);  
begin  
  CloseFile(ftxt);  
end;
```

Процедура построения списка

```
procedure TForm1.BuildList1Click(Sender: TObject);  
begin  
  BuildSpisok(PList,ftxt);  
end;
```

Процедура удаления списка

```
procedure TForm1.Dellist1Click(Sender: TObject);  
begin  
  DelSpisok(PList);  
end;
```

Процедура сохранения списка в текстовый файл

```
procedure TForm1.SaveListTxt1Click(Sender: TObject);  
begin  
  writeln(ftxt,'Содержимое двусвязанного списка ');  
  WriteSpText(PList,ftxt);
```

```
writeln(ftxt, ' ')  
end;
```

Процедура закрытия типизированного файла

```
procedure CloseTipFile (var fTip:FZap);  
begin  
  CloseFile(fTip);  
end;
```

Процедура вывода в типизированный файл

```
procedure WriteTipFile (var f:PUzel; var fTip: Fzap);  
var  
  p: PUzel;  
  y: Zak;  
begin  
  p:= f;  
  seek(ftip, filesize(ftip));  
  while not (p = nil) do  
    begin  
      y:= p^.x;  
      write (fTip,y);  
      p:= p^.next;  
    end;  
  end;
```

Процедура считывания с типизированного файла

```
procedure ReadTipFile (var f: PUzel; var fTip:Fzap);
```

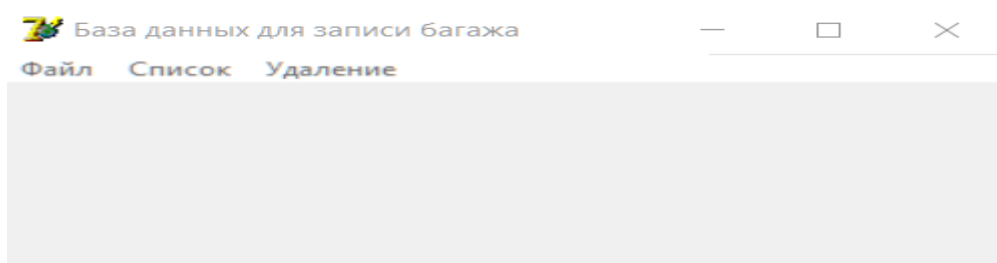
```

var
a,d:PUzel;
begin
if filesize(ftip)<>0 then // Если тип-ый файл не пустой, то создаём список
begin
seek(ftip,0);
begin
while Not(Eof(ftip)) do
begin
new(a);
read (fTip,a^.x);
a^.next:= nil;
a^.pred:=nil;
if(f=nil) then
begin
AddFirst (f,a);
d:=f;
end
else
begin
AddAfter(d,a);
d:=a;
end;
end;
end;
end;
end;
end;

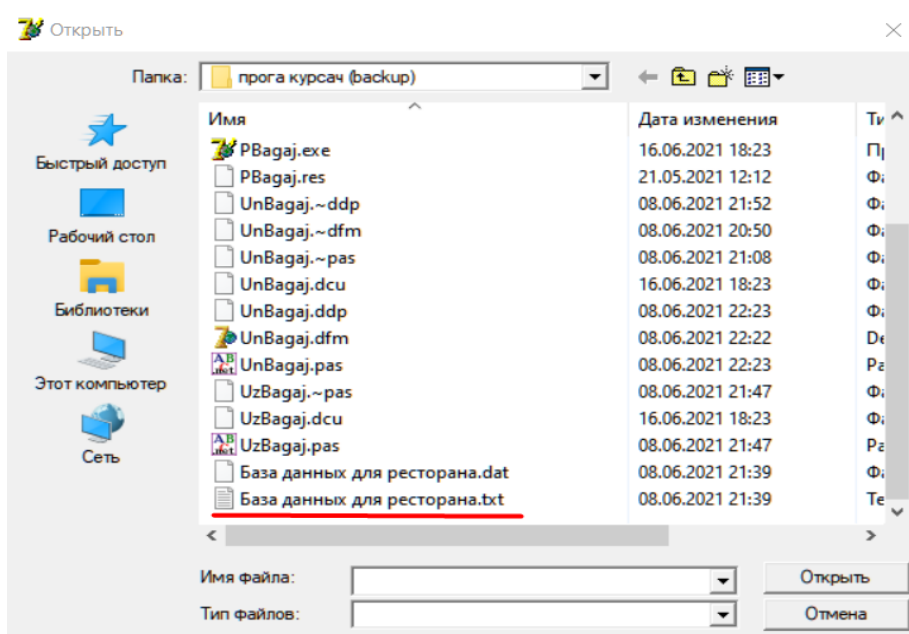
```

Глава 3. Тестирование приложения и проверка результатов

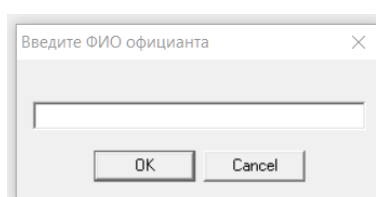
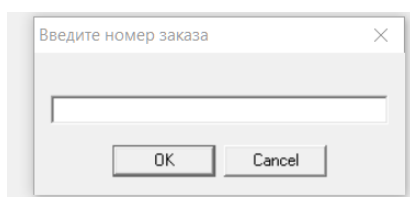
При запуске программы появляется главное окно приложения:

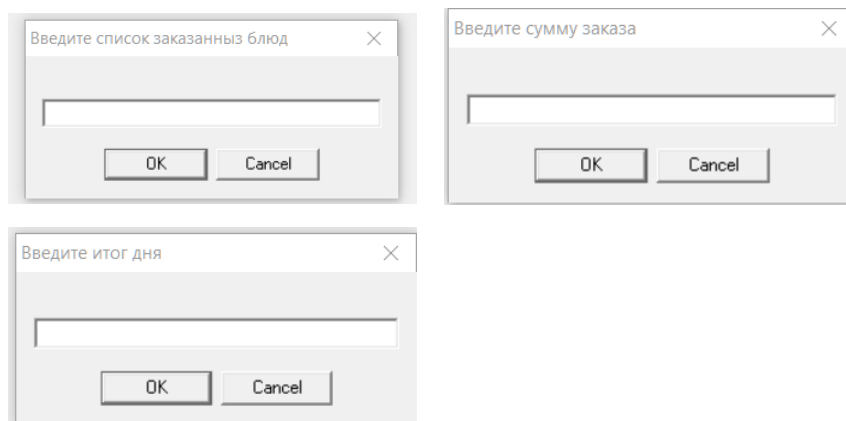


Для создания списка необходимо выполнить „Файл- открыть файл,,
Далее появится окно в котором нужно выбрать текстовый файл для записи в него результата.

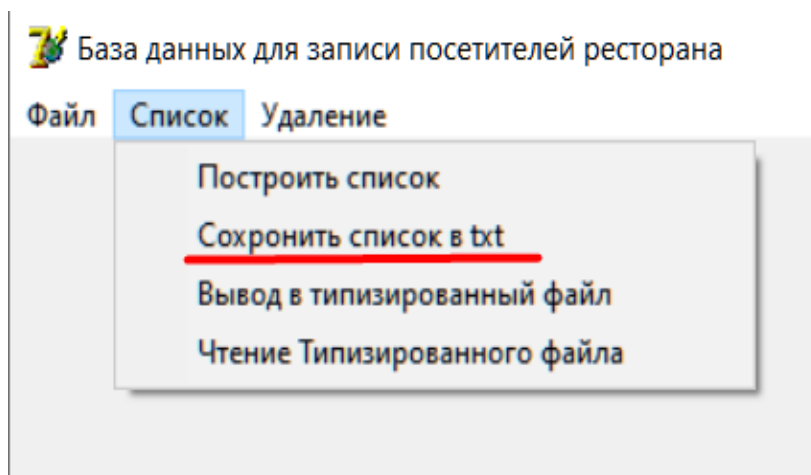


Для добавления данных в текстовый файл нужно нажать „Список- Построить список,, в появившееся окно ввести данные: номер заказа, ФИО официанта, список заказанных блюд, сумму заказа и итог дня.

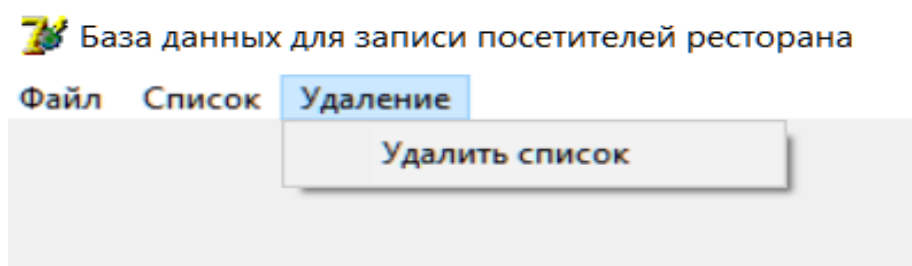




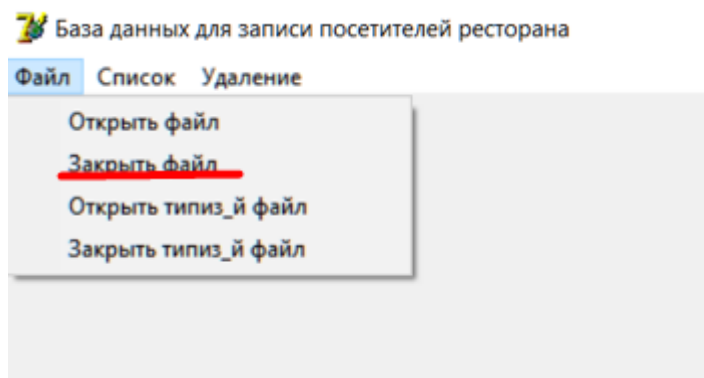
Для того что бы сохранить введенные данные в текстовый файл нажать ,,Список- Сохранить список в txt,,



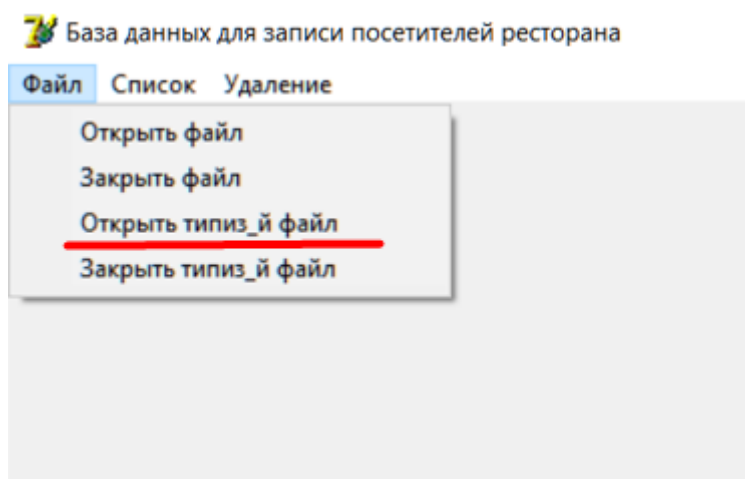
Для удаления списка нажмите ,,Удаление- Удалить список,,



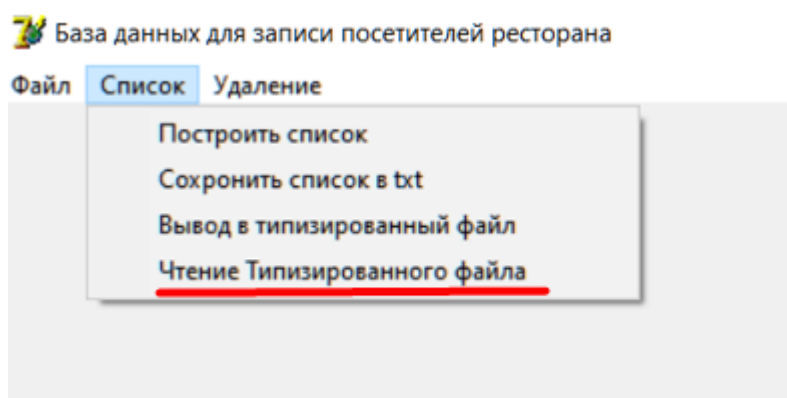
Для закрытия программы нажмите „Файл-Закрыть файл,,



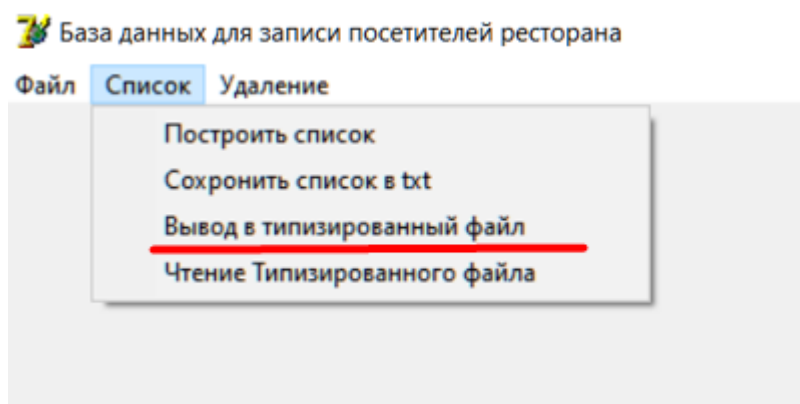
Для записи в типизированный файл нажмите „Файл-Открыть типизированный файл,,



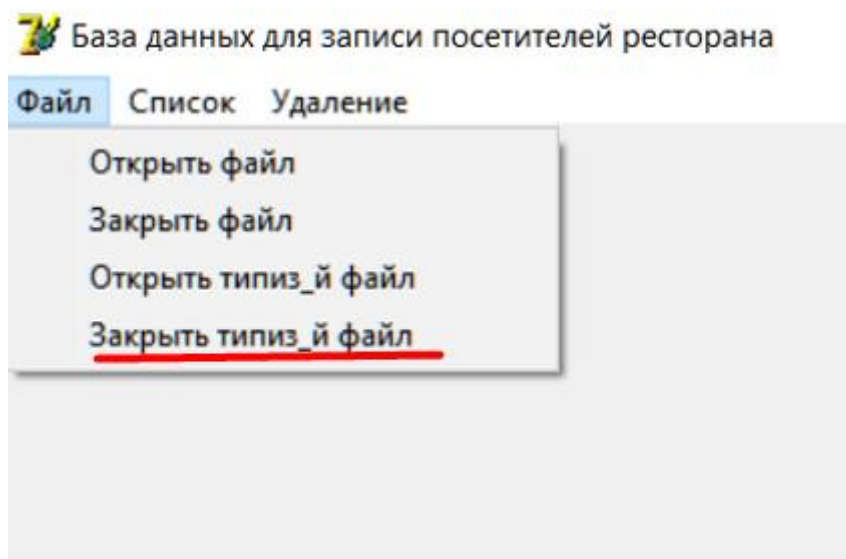
Для чтения из типизированного файла нажмите „Список- Чтение типизированного файла,,



Для вывода в типизированный файл нажмите „Список- Вывод в типизированный файл,,



Для закрытия типизированного файла нажмите „ Файл-Закрыть типизированный файл,,



Заключение

В курсовой работе были использованы материалы, изученные в течение двух семестров такие как: записи, типизированные и текстовые файлы, указатели, динамическая память, списковая структура, модули.

Так же были выполнены различные средства обработки баз данных, с помощью двусвязной списковой структуры:

- сохранение данных в файл (текстовый и типизированный);
- создание списка, а так же добавление данных в уже существующий список;
- удаление списка;
- загрузка списка из типизированного файла;

Представленные операции работают исправно, ошибки, полученные в процессе тестирования данной курсовой работы, были устранены.

Программа была протестирована порядка 10 раз. Руководство пользователя подробно описано с использованием скриншотов. В ходе работы было проанализировано четыре источника дополнительной литературы (таких как учебные пособия, учебники и вебсайты).

Литература

- 1) Компоненты Delphi [Электронный ресурс] / Режим доступа:
<http://www.delphi-manual.ru/lesson8.php>
- 2) Архангельский А.Я. Delphi 7. Справочное пособие: учебник – М.: Бином, 2004. – 1024 с.
- 3) Wikipedia База данных [Электронный ресурс] / Режим доступа:
https://ru.wikipedia.org/wiki/База_данных
- 4) Оформление курсовой работы [Электронный ресурс] / Общие требования к построению и оформлению текстовой документации ЗабГУ. – Режим доступа:
http://zabgu.ru/files/html_document/pdf_files/fixed/Normativny'e_do_kumenty'/MI__01-02-2018_Obshhie_trebovaniya_k_postroeniyu_i_oformleniyu_uchebnoj_tekstovoj_do_kumentacii.pdf

Приложения

Код интерфейсной части главной программы:

```
unit UnRestoranj;

interface

uses

  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, UzRestoran, Menus; type

  TForm1 = class(TForm)
    MainMenu: TMainMenu;
    OpenFileDialog: TOpenDialog;
    File1: TMenuItem;
    OpenTxtFile1: TMenuItem;
    CloseTxtFile1: TMenuItem;
    List1: TMenuItem;
    BuildList1: TMenuItem;
    SaveListTxt1: TMenuItem;
    Delete1: TMenuItem;
    DelList1: TMenuItem;
    procedure OpenTxtFile1Click(Sender: TObject);
    procedure CloseTxtFile1Click(Sender: TObject);
    procedure BuildList1Click(Sender: TObject);
    procedure DelList1Click(Sender: TObject);
    procedure SaveListTxt1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
```

```

end;

var
    Form1: TForm1;
    PList: PUzel; //указатель на список
    ftxt: TextFile; //файловая переменная

```

Код подключенного модуля:

```

unit UzRestoran;

interface

    uses SysUtils, Dialogs; //Модули для ввода/вывода данных

type
    Zak = record          //Тип заказа
        Number: real;      //номер стола
        FIO : string[64];  //Фиио официанта
        Zakaz: string [255]; //перечень блюд
        SumZak: real;      //сумма заказа
        ItogDay: real;     //итог дня
    end;

    PUzel = ^Zp;          //Тип указателя на узел
    Zp = record           //Тип узла списка
        x: Zak;            //Информация хранящаяся в узлах списка
        next: PUzel;       //Указатель на следующий узел
        pred: PUzel;       //Указатель на предыдущий узел
    end;

    FZap = file of Zak;   //Файловый тип для хранения данных о заказе

{Операции для работы с двусвязанным списком }

```

```

procedure AddFirst(var f: PUzel; a: PUzel);{Вставить узел а первым в список}
procedure AddAfter(var old:PUzel; a: PUzel);{Вставить узел а после old}
{Построить список; f -указатель на голову списка}
procedure BuildSpisok(var f: PUzel; var ftxt: TextFile);
{Вывод списка в текстовый файл}
procedure WriteSpText(var f: PUzel;var ftxt:Text);
{Выделить из списка первый узел и вернуть его пользователю}
procedure DelFirstElement(var f,a: PUzel);
{Выделить из списка узел.следующий за узлом old и вернуть его
пользователю}
procedure DelElement(var old,a: PUzel);
procedure DelSpisok(var f: PUzel); //Удалить список

```