Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №8**

**«ОСНОВЫ СОБЫТИЙНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ»**

**ПО МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ**

Выполнил: студент учебной группы

ИСПк-203-52-00

Сюткин Андрей Андреевич

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2023

**Цель работы:** получить представление об организации хранения и использования информации посредством типизированных файлов, закрепить навыки создания пользовательского интерфейса.

**Задание:**

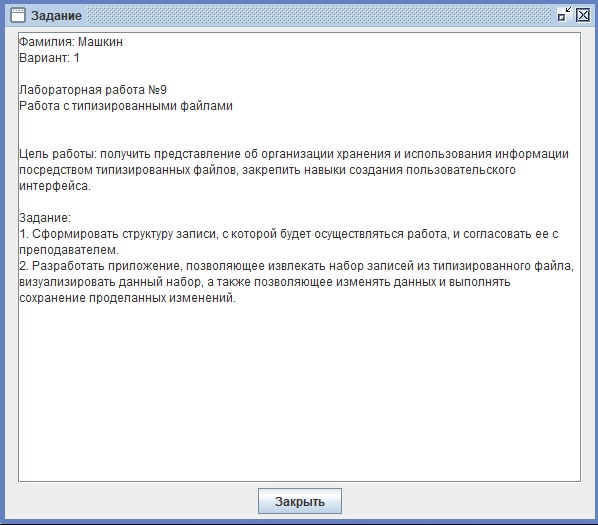


Рисунок 1 – Задание

Код программы:

unit sprav1\_unit1;

{$mode objfpc}{$H+}

interface

uses

Classes, edit\_unit1, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, Buttons,

Grids;

type

Contacts = record

Name: string[100];

Telephon: string[20];

Note: string[20];

end; //record

{ TfMain }

TfMain = class(TForm)

Panel1: TPanel;

bAdd: TSpeedButton;

bEdit: TSpeedButton;

bDel: TSpeedButton;

bSort: TSpeedButton;

SG: TStringGrid;

procedure bAddClick(Sender: TObject);

procedure bDelClick(Sender: TObject);

procedure bEditClick(Sender: TObject);

procedure bSortClick(Sender: TObject);

procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);

procedure FormCreate(Sender: TObject);

private

public

end;

var

fMain: TfMain;

adres: string; //адрес, откуда запущена программа

implementation

{$R \*.lfm}

{ TfMain }

procedure TfMain.bAddClick(Sender: TObject);

begin

//очищаем поля, если там что-то есть:

fEdit.eName.Text:= '';

fEdit.eTelephone.Text:= '';

//устанавливаем ModalResult редактора в mrNone:

fEdit.ModalResult:= mrNone;

//теперь выводим форму:

fEdit.ShowModal;

//если пользователь ничего не ввел - выходим:

if (fEdit.eName.Text= '') or (fEdit.eTelephone.Text= '') then exit;

//если пользователь не нажал "Сохранить" - выходим:

if fEdit.ModalResult <> mrOk then exit;

//иначе добавляем в сетку строку, и заполняем её:

SG.RowCount:= SG.RowCount + 1;

SG.Cells[0, SG.RowCount-1]:= fEdit.eName.Text;

SG.Cells[1, SG.RowCount-1]:= fEdit.eTelephone.Text;

SG.Cells[2, SG.RowCount-1]:= fEdit.CBNote.Text;

end;

procedure TfMain.bDelClick(Sender: TObject);

begin

//если данных нет - выходим:

if SG.RowCount = 1 then exit;

//иначе выводим запрос на подтверждение:

if MessageDlg('Требуется подтверждение',

'Вы действительно хотите удалить контакт "' +

SG.Cells[0, SG.Row] + '"?',

mtConfirmation, [mbYes, mbNo, mbIgnore], 0) = mrYes then

SG.DeleteRow(SG.Row);

end;

procedure TfMain.bEditClick(Sender: TObject);

begin

//если данных в сетке нет - просто выходим:

if SG.RowCount = 1 then exit;

//иначе записываем данные в форму редактора:

fEdit.eName.Text:= SG.Cells[0, SG.Row];

fEdit.eTelephone.Text:= SG.Cells[1, SG.Row];

fEdit.CBNote.Text:= SG.Cells[2, SG.Row];

//устанавливаем ModalResult редактора в mrNone:

fEdit.ModalResult:= mrNone;

//теперь выводим форму:

fEdit.ShowModal;

//сохраняем в сетку возможные изменения,

//если пользователь нажал "Сохранить":

if fEdit.ModalResult = mrOk then begin

SG.Cells[0, SG.Row]:= fEdit.eName.Text;

SG.Cells[1, SG.Row]:= fEdit.eTelephone.Text;

SG.Cells[2, SG.Row]:= fEdit.CBNote.Text;

end;

end;

procedure TfMain.bSortClick(Sender: TObject);

begin

//если данных в сетке нет - просто выходим:

if SG.RowCount = 1 then exit;

//иначе сортируем список:

SG.SortColRow(true, 0);

end;

procedure TfMain.FormClose(Sender: TObject; var CloseAction: TCloseAction);

var

MyCont: Contacts; //для очередной записи

f: file of Contacts; //файл данных

i: integer; //счетчик цикла

begin

//если строки данных пусты, просто выходим:

if SG.RowCount = 1 then exit;

//иначе открываем файл для записи:

try

AssignFile(f, adres + 'telephones.dat');

Rewrite(f);

//теперь цикл - от первой до последней записи сетки:

for i:= 1 to SG.RowCount-1 do begin

//получаем данные текущей записи:

MyCont.Name:= SG.Cells[0, i];

MyCont.Telephon:= SG.Cells[1, i];

MyCont.Note:= SG.Cells[2, i];

//записываем их:

Write(f, MyCont);

end;

finally

CloseFile(f);

end;

end;

procedure TfMain.FormCreate(Sender: TObject);

var

MyCont: Contacts; //для очередной записи

f: file of Contacts; //файл данных

i: integer; //счетчик цикла

begin

//сначала получим адрес программы:

adres:= ExtractFilePath(ParamStr(0));

//настроим сетку:

SG.Cells[0, 0]:= 'Модель';

SG.Cells[1, 0]:= 'Бренд';

SG.Cells[2, 0]:= 'Тип';

SG.ColWidths[0]:= 365;

SG.ColWidths[1]:= 150;

SG.ColWidths[2]:= 150;

//если файла данных нет, просто выходим:

if not FileExists(adres + 'telephones.dat') then exit;

//иначе файл есть, открываем его для чтения и

//считываем данные в сетку:

try

AssignFile(f, adres + 'telephones.dat');

Reset(f);

//теперь цикл - от первой до последней записи сетки:

while not Eof(f) do begin

//считываем новую запись:

Read(f, MyCont);

//добавляем в сетку новую строку, и заполняем её:

SG.RowCount:= SG.RowCount + 1;

SG.Cells[0, SG.RowCount-1]:= MyCont.Name;

SG.Cells[1, SG.RowCount-1]:= MyCont.Telephon;

SG.Cells[2, SG.RowCount-1]:= MyCont.Note;

end;

finally

CloseFile(f);

end;

end;

end.

unit edit\_unit1;

{$mode ObjFPC}{$H+}

interface

uses

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

type

{ TfEdit }

TfEdit = class(TForm)

bSave: TBitBtn;

bCancel: TBitBtn;

CBNote: TComboBox;

eName: TEdit;

eTelephone: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

procedure FormCreate(Sender: TObject);

procedure FormShow(Sender: TObject);

private

public

end;

var

fEdit: TfEdit;

implementation

{$R \*.lfm}

{ TfEdit }

procedure TfEdit.FormCreate(Sender: TObject);

begin

end;

procedure TfEdit.FormShow(Sender: TObject);

begin

eName.SetFocus;

end;

end.

**Результат выполнения программы:**



Рисунок 2 – Приложение Список вещей

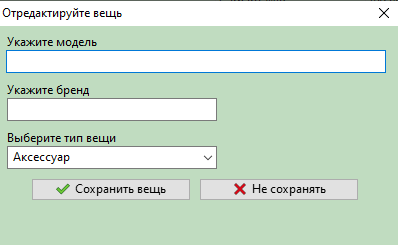


Рисунок 3 –Приложение Список вещей ч.2

**Вывод:** Нами была поставлена цель работы: получить представление об организации хранения и использования информации посредством типизированных файлов, закрепить навыки создания пользовательского интерфейса. Работа была разделена на несколько этапов: цель, формулировка задания, написание кода программы, результат выполнения программы, вывод. Код программы был написан на Lazarus. Итак, делая вывод, хочу сказать, что в ходе работы все цели были достигнуты.