Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №4**

**«РАБОТА С ТИПИЗИРОВАННЫМИ ФАЙЛАМИ»**

**ПО МДК 05.02 «РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ»**

Выполнил: студент учебной группы

ИСПк-202-52-00

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

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

**Задание**

1. Сформировать структуру записи, с которой будет осуществляться работа, и согласовать ее с преподавателем.

2. Разработать приложение, позволяющее извлекать набор записей из типизированного файла, визуализировать данный набор, а также позволяющее изменять данных и выполнять сохранение проделанных изменений.

## **1 Описание алгоритма**

Объекты, которые использовались при выполнении домашней контрольной работы:

TSpeedButton – кнопка для инструментальных панелей.

TPanel реализует унифицированный элемент управления. Панель используется для объединения нескольких компонентов.

TStringGrid позволяет отображать текстовые данные в ячейках, расположенных в строках и столбцах.

TLabel используется для отображения текста на экране.

TEdit – ввод и отображение строки.

TBitBtn используется для ввода команд в формы и диалоговые окна.

TComboBox используется для отображения выпадающего списка различных элементов.

## **2** **Код программы**

Модуль Main

unit main;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, ExtCtrls, Buttons,

Grids, edit;

type

{ 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;

type

Market = record

Name: string[100];

Price: string[20];

Warehouse: string[10];

Maker: string[100];

Category: string[100];

end; //record

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.ePrice.Text:= '';

fEdit.eCategory.Text:= '';

fEdit.eMaker.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.ePrice.Text = '') or (fEdit.eCategory.Text = '') or (fEdit.eMaker.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.ePrice.Text;

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

SG.Cells[3, SG.RowCount-1]:= fEdit.CBNote.Text;

SG.Cells[4, SG.RowCount-1]:= fEdit.eCategory.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 товар "' +

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

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

fEdit.CBNote.Text:= SG.Cells[3, SG.Row];

fEdit.eCategory.Text:= SG.Cells[4, 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.ePrice.Text;

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

SG.Cells[3, SG.Row]:= fEdit.CBNote.Text;

SG.Cells[4, SG.Row]:= fEdit.eCategory.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

MyMarket: Market; //для очередной записи

f: file of Market; //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:

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

MyMarket.Price:= SG.Cells[1, i];

MyMarket.Warehouse:= SG.Cells[2, i];

MyMarket.Maker:= SG.Cells[3, i];

MyMarket.Category:= SG.Cells[4, i];

//записываем их:

Write(f, MyMarket);

end;

finally

CloseFile(f);

end;

end;

procedure TfMain.FormCreate(Sender: TObject);

var

MyMarket: Market; //для очередной записи

f: file of Market; //файл данных

i: integer; //счетчик цикла

begin

//сначала получим адрес программы:

adres:= ExtractFilePath(ParamStr(0));

//настроим сетку:

SG.Cells[0, 0]:= 'Наименование';

SG.Cells[1, 0]:= 'Цена';

SG.Cells[2, 0]:= 'Производитель';

SG.Cells[3, 0]:= 'Наличие на складе';

SG.Cells[4, 0]:= 'Категория товара';

SG.ColWidths[0]:= 150;

SG.ColWidths[1]:= 100;

SG.ColWidths[2]:= 150;

SG.ColWidths[3]:= 150;

SG.ColWidths[4]:= 250;

//если файла данных нет, просто выходим:

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

//иначе файл есть, открываем его для чтения и

//считываем данные в сетку:

try

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

Reset(f);

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

while not Eof(f) do begin

//считываем новую запись:

Read(f, MyMarket);

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

SG.RowCount:= SG.RowCount + 1;

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

SG.Cells[1, SG.RowCount-1]:= MyMarket.Price;

SG.Cells[2, SG.RowCount-1]:= MyMarket.Warehouse;

SG.Cells[3, SG.RowCount-1]:= MyMarket.Maker;

SG.Cells[4, SG.RowCount-1]:= MyMarket.Category;

end;

finally

CloseFile(f);

end;

end;

end.

Модуль Edit

unit edit;

{$mode ObjFPC}{$H+}

interface

uses

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

type

{ TfEdit }

TfEdit = class(TForm)

bSave: TBitBtn;

bCancel: TBitBtn;

CBNote: TComboBox;

eCategory: TComboBox;

eName: TEdit;

ePrice: TEdit;

eMaker: TEdit;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

procedure FormShow(Sender: TObject);

private

public

end;

var

fEdit: TfEdit;

implementation

{$R \*.lfm}

{ TfEdit }

procedure TfEdit.FormShow(Sender: TObject);

begin

eName.SetFocus;

end;

end.

**3 Результат выполнения программы**

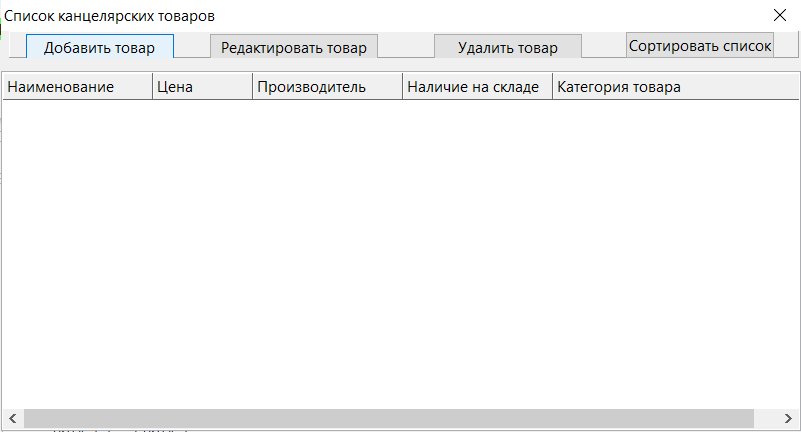


Рисунок 2 – Результат выполнения программы

## **4 Вывод**

В ходе выполнения данной контрольной работы было получено представление об организации хранения и использования информации посредством типизированных файлов, были закреплены навыки создания пользовательского интерфейса.