УО «Белорусский государственный университет информатики и

радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №4.1

по предмету

Основы Алгоритмизации и Программирования

Вариант 4

Выполнил

Воривода М.А.

Проверила

Данилова Г.В.

Группа:

951007

Минск 2020

**Задание**

Дана матрица a(m,n). Найти в ней путь от элемента a[i1,j1] до элемента a[i2,j2] с максимальной суммой. Ходить можно по горизонталям и вертикалям. Каждый элемент матрицы может входить в путь не более двух раз.

**Код программы**

**(Delphi)**

unit Unit1;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.Menus, Vcl.StdCtrls;

type

TForm1 = class(TForm)

MainMenu1: TMainMenu;

File1: TMenuItem;

Help1: TMenuItem;

Label1: TLabel;

Label2: TLabel;

Timetable: TButton;

Exit: TButton;

Help: TButton;

Edit: TButton;

Find: TButton;

View: TButton;

Create: TButton;

Create1: TMenuItem;

View1: TMenuItem;

Find1: TMenuItem;

Edit1: TMenuItem;

N1: TMenuItem;

Exit1: TMenuItem;

Hotkeys1: TMenuItem;

Aboutprogram1: TMenuItem;

Aboutauthor1: TMenuItem;

N2: TMenuItem;

PopupMenu1: TPopupMenu;

procedure FormCloseQuery(Sender: TObject; var CanClose: Boolean);

procedure ExitClick(Sender: TObject);

procedure Hotkeys1Click(Sender: TObject);

procedure Aboutprogram1Click(Sender: TObject);

procedure Aboutauthor1Click(Sender: TObject);

procedure HelpClick(Sender: TObject);

procedure CreateClick(Sender: TObject);

procedure FormMoving(var m: TMessage); message WM\_MOVING;

procedure ViewClick(Sender: TObject);

procedure FindClick(Sender: TObject);

procedure EditClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

Form1: TForm1;

OpenedWindow: Char;

implementation

{$R \*.dfm}

uses Unit2, Unit3, Unit4, Unit5;

procedure TForm1.CreateClick(Sender: TObject);

begin

case OpenedWindow of

'V':

ViewForm.Close;

'F':

FindForm.Close;

'E':

EditForm.Close;

end;

CreateForm.Left := Form1.Left + 4;

CreateForm.Top := Form1.Top + 110;

CreateForm.Show;

OpenedWindow := 'C';

end;

procedure TForm1.EditClick(Sender: TObject);

begin

case OpenedWindow of

'V':

ViewForm.Close;

'F':

FindForm.Close;

'C':

CreateForm.Close;

end;

EditForm.Left := Form1.Left + 4;

EditForm.Top := Form1.Top + 110;

EditForm.Show;

OpenedWindow := 'E';

end;

procedure TForm1.FindClick(Sender: TObject);

begin

case OpenedWindow of

'V':

ViewForm.Close;

'C':

CreateForm.Close;

'E':

EditForm.Close;

end;

FindForm.Left := Form1.Left + 4;

FindForm.Top := Form1.Top + 110;

FindForm.Show;

OpenedWindow := 'F';

end;

procedure TForm1.ViewClick(Sender: TObject);

begin

case OpenedWindow of

'C':

CreateForm.Close;

'F':

FindForm.Close;

'E':

EditForm.Close;

end;

ViewForm.Left := Form1.Left + 4;

ViewForm.Top := Form1.Top + 110;

ViewForm.Show;

OpenedWindow := 'V';

end;

procedure TForm1.ExitClick(Sender: TObject);

begin

Form1.Close;

end;

procedure TForm1.FormCloseQuery(Sender: TObject; var CanClose: Boolean);

begin

if MessageBox(0, 'Do you want to exit?', 'Exit',

MB\_YESNO + MB\_ICONWARNING) = 6 then

CanClose := true

else

CanClose := false;

end;

procedure TForm1.FormMoving(var m: TMessage);

begin

case OpenedWindow of

'C':

begin

CreateForm.Left := Form1.Left + 4;

CreateForm.Top := Form1.Top + 110;

end;

'V':

begin

ViewForm.Left := Form1.Left + 4;

ViewForm.Top := Form1.Top + 110;

end;

'F':

begin

FindForm.Left := Form1.Left + 4;

FindForm.Top := Form1.Top + 110;

end;

'E':

begin

EditForm.Left := Form1.Left + 4;

EditForm.Top := Form1.Top + 110;

end;

end;

end;

procedure TForm1.HelpClick(Sender: TObject);

begin

MessageBox(0,

' This program provides an opportunity to create, view and edit timetable of classes. '

+ 'There is hot keys in application which simplifies using of program.' +

#13#10 + ' You in main window of app. To use other functions ' +

'click on relevant buttons. More information in "Help".', 'About program',

MB\_OK + MB\_ICONINFORMATION);

end;

procedure TForm1.Hotkeys1Click(Sender: TObject);

begin

MessageBox(0, 'Alt+C - create timetable' + #13#10 +

'Alt+V - view timetable' + #13#10 +

'Alt+F - search in timetable' + #13#10 +

'Alt+E - edit timetable' + #13#10 + 'Esc - exit' + #13#10 +

'Alt+H - hot keys' + #13#10 + 'Alt+P - about program' + #13#10 +

'Alt+A - about author', 'Hot keys', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TForm1.Aboutauthor1Click(Sender: TObject);

begin

MessageBox(0, 'Created by Matvey Vorivoda, student of group 951007.',

'About author', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TForm1.Aboutprogram1Click(Sender: TObject);

begin

MessageBox(0,

'This program provides an opportunity to create, view and edit timetable of classes. There is hot keys in application which simplifies using of program.',

'About program', MB\_OK + MB\_ICONINFORMATION);

end;

end.

unit Unit2;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Menus, Vcl.Grids,

Vcl.ExtCtrls, Vcl.ExtDlgs;

type

TTimetable = record

Professor, Day, Order, Subject: String[100];

end;

TCreateForm = class(TForm)

ProfessorEdit: TEdit;

SubjectEdit: TEdit;

DayEdit: TComboBox;

OrderEdit: TComboBox;

PopupMenu1: TPopupMenu;

AddRecordButton: TButton;

Timetable: TStringGrid;

SaveButton: TButton;

ClearButton: TButton;

HelpButton: TButton;

CloseButton: TButton;

Panel1: TPanel;

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

MainMenu1: TMainMenu;

File1: TMenuItem;

Save1: TMenuItem;

Clear1: TMenuItem;

N1: TMenuItem;

Close1: TMenuItem;

About1: TMenuItem;

Help1: TMenuItem;

Hotkeys1: TMenuItem;

SaveTextFileDialog1: TSaveTextFileDialog; procedure DayKeyPress(Sender: TObject; var key: Char);

procedure OrderKeyPress(Sender: TObject; var key: Char);

procedure OrderExit(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure AddRecordButtonClick(Sender: TObject);

procedure ProfessorKeyPress(Sender: TObject; var key: Char);

procedure SubjectKeyPress(Sender: TObject; var key: Char);

procedure CloseButtonClick(Sender: TObject);

procedure ClearButtonClick(Sender: TObject);

procedure HelpButtonClick(Sender: TObject);

procedure Hotkeys1Click(Sender: TObject);

procedure Save1Click(Sender: TObject);

procedure GridClear();

private

{ Private declarations }

public

{ Public declarations }

end;

var

CreateForm: TCreateForm;

implementation

{$R \*.dfm}

procedure TCreateForm.FormCreate(Sender: TObject);

begin

Timetable.Cells[0, 0] := 'Professor';

Timetable.Cells[1, 0] := 'Day';

Timetable.Cells[2, 0] := 'Order';

Timetable.Cells[3, 0] := 'Subject';

Timetable.ColWidths[0] := 250;

Timetable.ColWidths[1] := 50;

Timetable.ColWidths[2] := 50;

Timetable.ColWidths[3] := 223;

end;

procedure TCreateForm.GridClear;

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

end;

procedure TCreateForm.HelpButtonClick(Sender: TObject);

begin

MessageBox(0,

' In this window you should fill every field and click on "Add' +

' record" to add new record to timetable. Then you can save the timetable'

+ #13#10 + ' Also you can clear timetable and close this window.',

'Help', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TCreateForm.Hotkeys1Click(Sender: TObject);

begin

MessageBox(0, 'Alt+S - save timetable' + #13#10 + 'Alt+C - clear timetable'

+ #13#10 + 'Esc - close window' + #13#10 + 'Alt+E - help' + #13#10 +

'Alt+H - hot keys', 'Hot keys', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TCreateForm.Save1Click(Sender: TObject);

var

Put : file of TTimetable;

RowRecord : TTimetable;

I : Integer;

begin

if SaveTextFileDialog1.Execute then

try

AssignFile(Put, SaveTextFileDialog1.FileName);

Rewrite(Put);

with Timetable do

for I := 1 to RowCount - 1 do

begin

RowRecord.Professor := Cells[0, I];

RowRecord.Day := Cells[1, I];

RowRecord.Order := Cells[2, I];

RowRecord.Subject := Cells[3, I];

Write(Put, RowRecord);

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error', MB\_OK + MB\_ICONERROR);

end;

end;

procedure TCreateForm.SubjectKeyPress(Sender: TObject; var key: Char);

begin

if key = #13 then

AddRecordButtonClick(AddRecordButton);

if Length(SubjectEdit.Text + key) > 100 then key := #0;

end;

procedure TCreateForm.ProfessorKeyPress(Sender: TObject; var key: Char);

begin

if Length(ProfessorEdit.Text + key) > 100 then key := #0;

case key of

#8:

;

#13:

SubjectEdit.SetFocus;

'A' .. 'Z':

;

'a' .. 'z':

;

'а' .. 'я':

;

'А' .. 'Я':

;

else

key := #0;

end;

end;

procedure TCreateForm.AddRecordButtonClick(Sender: TObject);

begin

if (ProfessorEdit.Text = '') or (SubjectEdit.Text = '') then

MessageBox(0, 'Fill every field', 'Error', MB\_OK + MB\_ICONWARNING)

else

begin

with Timetable do

begin

Cells[0, RowCount - 1] := ProfessorEdit.Text;

Cells[1, RowCount - 1] := DayEdit.Text;

Cells[2, RowCount - 1] := OrderEdit.Text;

Cells[3, RowCount - 1] := SubjectEdit.Text;

RowCount := RowCount + 1;

ProfessorEdit.Text := '';

DayEdit.Text := 'Mon';

OrderEdit.Text := '1';

SubjectEdit.Text := '';

end;

ProfessorEdit.SetFocus;

end;

end;

procedure TCreateForm.ClearButtonClick(Sender: TObject);

begin

ProfessorEdit.Text := '';

SubjectEdit.Text := '';

OrderEdit.Text := '1';

DayEdit.Text := 'Mon'; {}

GridClear();

end;

procedure TCreateForm.CloseButtonClick(Sender: TObject);

begin

ProfessorEdit.Text := '';

SubjectEdit.Text := '';

OrderEdit.Text := '1';

DayEdit.Text := 'Mon';

GridClear();

CreateForm.Close; {}

end;

procedure TCreateForm.DayKeyPress(Sender: TObject; var key: Char);

begin

key := #0;

end;

procedure TCreateForm.OrderKeyPress(Sender: TObject; var key: Char);

begin

case key of

#8:

;

'1' .. '5':

if Length((Sender as TComboBox).Text + key) > 1 then

key := #0;

else

key := #0;

end;

end;

procedure TCreateForm.OrderExit(Sender: TObject);

begin

if OrderEdit.Text = '' then

OrderEdit.Text := '1';

end;

end.

unit Unit3;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Menus, Vcl.ExtCtrls,

Vcl.Grids, Vcl.ExtDlgs;

type

TTimetable = record

Professor, Day, Order, Subject: String[100];

end;

TViewForm = class(TForm)

Timetable: TStringGrid;

MainMenu1: TMainMenu;

File1: TMenuItem;

Open1: TMenuItem;

N1: TMenuItem;

Close1: TMenuItem;

About1: TMenuItem;

Help1: TMenuItem;

Hotkeys1: TMenuItem;

Panel1: TPanel;

PopupMenu1: TPopupMenu;

OpenButton: TButton;

HelpButton: TButton;

CloseButton: TButton;

OpenTextFileDialog1: TOpenTextFileDialog;

procedure FormCreate(Sender: TObject);

procedure Close1Click(Sender: TObject);

procedure Help1Click(Sender: TObject);

procedure Hotkeys1Click(Sender: TObject);

procedure Open1Click(Sender: TObject);

procedure TimetableSort(Method: Integer);

procedure TimetableSelectCell(Sender: TObject; ACol, ARow: Integer;

var CanSelect: Boolean);

procedure NonDaySort(Method: Integer);

procedure RowSwop(First, Second: Integer);

procedure GridClear;

procedure DaySort;

private

{ Private declarations }

public

{ Public declarations }

end;

var

ViewForm: TViewForm;

implementation

{$R \*.dfm}

uses Unit6, Unit2;

procedure TViewForm.Close1Click(Sender: TObject);

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

ViewForm.Close;

end;

procedure TViewForm.RowSwop(First: Integer; Second: Integer); { СОРТИРОВКА }

var

Temp: String;

I: Integer;

begin

with Timetable do

for I := 0 to 3 do

begin

Temp := Cells[I, First];

Cells[I, First] := Cells[I, Second];

Cells[I, Second] := Temp;

end;

end;

procedure TViewForm.GridClear;

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

end;

procedure TViewForm.NonDaySort(Method: Integer);

var

I, J: Integer;

begin

with Timetable do

begin

for I := 1 to RowCount - 1 do

for J := 1 to RowCount - I - 1 do

if Cells[Method, J] > Cells[Method, J + 1] then

RowSwop(J, J + 1);

end;

end; { СОРТИРОВКА }

procedure TViewForm.DaySort();

var

I, J, Current : Integer;

begin

Current := 1;

with Timetable do

begin

for I := 0 to 5 do

for J := 1 to RowCount - 1 do

if Cells[1, J] = CreateForm.DayEdit.Items[I] then

begin

RowSwop(Current, J);

Inc(Current);

end;

end;

end; { СОРТИРОВКА }

procedure TViewForm.TimetableSort(Method: Integer);

begin

case Method of

0, 2, 3:

NonDaySort(Method);

1: DaySort();

end;

end; { СОРТИРОВКА }

procedure TViewForm.TimetableSelectCell(Sender: TObject; ACol: Integer;

ARow: Integer; var CanSelect: Boolean);

begin

if ARow = 0 then

TimetableSort(ACol)

else

begin

with Timetable do

begin

ModalView.ProfessorEdit.Text := Cells[0, ARow];

ModalView.DayEdit.Text := Cells[1, ARow];

ModalView.OrderEdit.Text := Cells[2, ARow];

ModalView.SubjectEdit.Text := Cells[3, ARow];

ModalView.ShowModal;

end;

end;

end; { СОРТИРОВКА }

procedure TViewForm.FormCreate(Sender: TObject);

begin

with Timetable do

begin

Cells[0, 0] := 'Professor';

Cells[1, 0] := 'Day';

Cells[2, 0] := 'Order';

Cells[3, 0] := 'Subject';

ColWidths[0] := 250;

ColWidths[1] := 50;

ColWidths[2] := 50;

ColWidths[3] := 223;

Row := 1;

Col := 0;

end;

end;

procedure TViewForm.Help1Click(Sender: TObject);

begin

MessageBox(0, ' This window provides viewing of any saved timetable.' +

' There is hot keys which simplifies using of program.' + #13#10 +

' You can sort timetable by click on title of column.', 'Help',

MB\_OK + MB\_ICONINFORMATION);

end;

procedure TViewForm.Hotkeys1Click(Sender: TObject);

begin

MessageBox(0, 'Alt+O - open timetable' + #13#10 + 'Alt+E - help' + #13#10 +

'Alt+H - hot keys' + #13#10 + 'Esc - close', 'Hot keys',

MB\_OK + MB\_ICONINFORMATION);

end;

procedure TViewForm.Open1Click(Sender: TObject);

var

Put: file of TTimetable;

I: Integer;

RowRecord: TTimetable;

begin

I := 1;

GridClear();

if OpenTextFileDialog1.Execute then

try

AssignFile(Put, OpenTextFileDialog1.FileName);

Reset(Put);

if EoF(Put) then

MessageBox(0, 'Error! File is empty.', 'File error',

MB\_OK + MB\_ICONERROR)

else

begin

with Timetable do

begin

while not EoF(Put) do

try

Read(Put, RowRecord);

Cells[0, I] := RowRecord.Professor;

Cells[1, I] := RowRecord.Day;

Cells[2, I] := RowRecord.Order;

Cells[3, I] := RowRecord.Subject;

RowCount := RowCount + 1;

Inc(I);

except

MessageBox(0,

PWideChar('Error! Error on step №' + IntToStr(I) +

'.'), 'Error', MB\_OK + MB\_ICONERROR);

end;

RowCount := RowCount - 2;

end;

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error',

MB\_OK + MB\_ICONERROR);

end;

end;

end.

unit Unit4;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.ExtCtrls, Vcl.Grids,

Vcl.Menus, Vcl.ExtDlgs;

type

TTimetable = record

Professor, Day, Order, Subject: String[100];

end;

TFindForm = class(TForm)

OpenTextFileDialog1: TOpenTextFileDialog;

SaveTextFileDialog1: TSaveTextFileDialog;

PopupMenu1: TPopupMenu;

MainMenu1: TMainMenu;

File1: TMenuItem;

Open1: TMenuItem;

Saveas1: TMenuItem;

N1: TMenuItem;

Close1: TMenuItem;

About1: TMenuItem;

Help1: TMenuItem;

Hotkeys1: TMenuItem;

Timetable: TStringGrid;

Panel1: TPanel;

Label1: TLabel;

Label2: TLabel;

CritEdit: TComboBox;

FindEdit: TEdit;

SearchButton: TButton;

Button1: TButton;

Button2: TButton;

Button3: TButton;

Button5: TButton;

procedure FormCreate(Sender: TObject);

procedure Close1Click(Sender: TObject);

procedure GridClear;

procedure Help1Click(Sender: TObject);

procedure Hotkeys1Click(Sender: TObject);

procedure Open1Click(Sender: TObject);

procedure Saveas1Click(Sender: TObject);

procedure CritEditKeyPress(Sender: TObject; var key: Char);

procedure FindEditKeyPress(Sender: TObject; var key: Char);

procedure SearchButtonClick(Sender: TObject);

procedure Search();

procedure RowTrans(First: Integer; Second: Integer);

private

{ Private declarations }

public

{ Public declarations }

end;

var

FindForm: TFindForm;

implementation

{$R \*.dfm}

procedure TFindForm.Close1Click(Sender: TObject);

begin

Timetable.RowCount := 2;

CritEdit.ItemIndex := 0;

FindEdit.Text := '';

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

FindForm.Close;

end;

procedure TFindForm.CritEditKeyPress(Sender: TObject; var key: Char);

begin

key := #0;

end;

procedure TFindForm.FormCreate(Sender: TObject);

begin

with Timetable do

begin

Cells[0, 0] := 'Professor';

Cells[1, 0] := 'Day';

Cells[2, 0] := 'Order';

Cells[3, 0] := 'Subject';

ColWidths[0] := 250;

ColWidths[1] := 50;

ColWidths[2] := 50;

ColWidths[3] := 223;

Row := 1;

Col := 0;

end;

end;

procedure TFindForm.Help1Click(Sender: TObject);

begin

MessageBox(0, ' In this window you can find any record' +

' in timetable by selected criterion', 'Help',

MB\_OK + MB\_ICONINFORMATION);

end;

procedure TFindForm.Hotkeys1Click(Sender: TObject);

begin

MessageBox(0, 'Alt+O - open timetable' + #13#10 + 'Alt+S - save timetable' +

#13#10 + 'Alt+E - help' + #13#10 + 'Alt+H - hot keys' + #13#10 +

'Esc - close', 'Hot keys', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TFindForm.Open1Click(Sender: TObject);

var

Put: file of TTimetable;

I: Integer;

RowRecord: TTimetable;

begin

I := 1;

GridClear();

if OpenTextFileDialog1.Execute then

try

AssignFile(Put, OpenTextFileDialog1.FileName);

Reset(Put);

if EoF(Put) then

MessageBox(0, 'Error! File is empty.', 'File error',

MB\_OK + MB\_ICONERROR)

else

begin

with Timetable do

begin

while not EoF(Put) do

try

Read(Put, RowRecord);

Cells[0, I] := RowRecord.Professor;

Cells[1, I] := RowRecord.Day;

Cells[2, I] := RowRecord.Order;

Cells[3, I] := RowRecord.Subject;

RowCount := RowCount + 1;

Inc(I);

except

MessageBox(0,

PWideChar('Error! Error on step №' + IntToStr(I) +

'.'), 'Error', MB\_OK + MB\_ICONERROR);

end;

RowCount := RowCount - 2;

end;

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error',

MB\_OK + MB\_ICONERROR);

end;

end;

procedure TFindForm.Saveas1Click(Sender: TObject);

var

Put: file of TTimetable;

RowRecord: TTimetable;

I: Integer;

begin

if SaveTextFileDialog1.Execute then

try

AssignFile(Put, SaveTextFileDialog1.FileName);

Rewrite(Put);

with Timetable do

for I := 1 to RowCount - 1 do

begin

RowRecord.Professor := Cells[0, I];

RowRecord.Day := Cells[1, I];

RowRecord.Order := Cells[2, I];

RowRecord.Subject := Cells[3, I];

Write(Put, RowRecord);

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error',

MB\_OK + MB\_ICONERROR);

end;

end;

procedure TFindForm.GridClear;

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

end;

procedure TFindForm.SearchButtonClick(Sender: TObject);

begin

Search();

end;

procedure TFindForm.Search;

var

I: Integer;

Goals: Array of Integer;

begin

SetLength(Goals, 0);

with Timetable do

begin

for I := 1 to RowCount - 1 do

if Cells[CritEdit.ItemIndex, I] = FindEdit.Text then

begin

SetLength(Goals, Length(Goals) + 1);

Goals[High(Goals)] := I;

end;

for I := 1 to Length(Goals) do

RowTrans(I, Goals[I - 1]);

RowCount := Length(Goals) + 1;

end;

end; {}

procedure TFindForm.RowTrans(First: Integer; Second: Integer); { СОРТИРОВКА }

var

I: Integer;

begin

with Timetable do

for I := 0 to 3 do

Cells[I, First] := Cells[I, Second];

end;

procedure TFindForm.FindEditKeyPress(Sender: TObject; var key: Char);

begin

if key = #13 then

SearchButtonClick(SearchButton);

if Length(FindEdit.Text + key) > 100 then

key := #0;

end;

end.

unit Unit5;

interface

uses

Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants,

System.Classes, Vcl.Graphics,

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.ExtDlgs, Vcl.Menus, Vcl.ExtCtrls,

Vcl.StdCtrls, Vcl.Grids;

type

TTimetable = record

Professor, Day, Order, Subject: String[100];

end;

TEditForm = class(TForm)

Timetable: TStringGrid;

Button5: TButton;

Button3: TButton;

Button2: TButton;

Button1: TButton;

Panel1: TPanel;

PopupMenu1: TPopupMenu;

MainMenu1: TMainMenu;

File1: TMenuItem;

Open1: TMenuItem;

Saveas1: TMenuItem;

N1: TMenuItem;

Close1: TMenuItem;

About1: TMenuItem;

Help1: TMenuItem;

Hotkeys1: TMenuItem;

SaveTextFileDialog1: TSaveTextFileDialog;

OpenTextFileDialog1: TOpenTextFileDialog;

procedure Open1Click(Sender: TObject);

procedure Saveas1Click(Sender: TObject);

procedure Close1Click(Sender: TObject);

procedure Help1Click(Sender: TObject);

procedure Hotkeys1Click(Sender: TObject);

procedure GridClear;

procedure FormCreate(Sender: TObject);

procedure TimetableSelectCell(Sender: TObject; ACol: Integer;

ARow: Integer; var CanSelect: Boolean);

procedure TimetableSort(Method: Integer);

procedure RowSwop(First: Integer; Second: Integer);

procedure NonDaySort(Method: Integer);

procedure DaySort();

private

{ Private declarations }

public

{ Public declarations }

end;

var

EditForm: TEditForm;

implementation

{$R \*.dfm}

uses Unit7, Unit2;

procedure TEditForm.Close1Click(Sender: TObject);

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

EditForm.Close;

end;

procedure TEditForm.TimetableSelectCell(Sender: TObject; ACol: Integer;

ARow: Integer; var CanSelect: Boolean);

begin

if ARow = 0 then

TimetableSort(ACol)

else

begin

with Timetable do

begin

ModalEdit.OpenedIndex := ARow;

ModalEdit.ProfessorEdit.Text := Cells[0, ARow];

ModalEdit.DayEdit.Text := Cells[1, ARow];

ModalEdit.OrderEdit.Text := Cells[2, ARow];

ModalEdit.SubjectEdit.Text := Cells[3, ARow];

ModalEdit.ShowModal;

end;

end;

end;

procedure TEditForm.TimetableSort(Method: Integer);

begin

case Method of

0, 2, 3:

NonDaySort(Method);

1:

DaySort();

end;

end;

procedure TEditForm.RowSwop(First: Integer; Second: Integer); { СОРТИРОВКА }

var

Temp: String;

I: Integer;

begin

with Timetable do

for I := 0 to 3 do

begin

Temp := Cells[I, First];

Cells[I, First] := Cells[I, Second];

Cells[I, Second] := Temp;

end;

end;

procedure TEditForm.NonDaySort(Method: Integer);

var

I, J: Integer;

begin

with Timetable do

begin

for I := 1 to RowCount - 1 do

for J := 1 to RowCount - I - 1 do

if Cells[Method, J] > Cells[Method, J + 1] then

RowSwop(J, J + 1);

end;

end;

procedure TEditForm.DaySort();

var

I, J, Current: Integer;

begin

Current := 1;

with Timetable do

begin

for I := 0 to 5 do

for J := 1 to RowCount - 1 do

if Cells[1, J] = CreateForm.DayEdit.Items[I] then

begin

RowSwop(Current, J);

Inc(Current);

end;

end;

end;

procedure TEditForm.Help1Click(Sender: TObject);

begin

MessageBox(0, ' In this window you can find any record' +

' in timetable by selected criterion', 'Help',

MB\_OK + MB\_ICONINFORMATION);

end;

procedure TEditForm.Hotkeys1Click(Sender: TObject);

begin

MessageBox(0, 'Alt+O - open timetable' + #13#10 + 'Alt+S - save timetable' +

#13#10 + 'Alt+E - help' + #13#10 + 'Alt+H - hot keys' + #13#10 +

'Esc - close', 'Hot keys', MB\_OK + MB\_ICONINFORMATION);

end;

procedure TEditForm.Open1Click(Sender: TObject);

var

Put: file of TTimetable;

I: Integer;

RowRecord: TTimetable;

begin

I := 1;

GridClear();

if OpenTextFileDialog1.Execute then

try

AssignFile(Put, OpenTextFileDialog1.FileName);

Reset(Put);

if EoF(Put) then

MessageBox(0, 'Error! File is empty.', 'File error',

MB\_OK + MB\_ICONERROR)

else

begin

with Timetable do

begin

while not EoF(Put) do

try

Read(Put, RowRecord);

Cells[0, I] := RowRecord.Professor;

Cells[1, I] := RowRecord.Day;

Cells[2, I] := RowRecord.Order;

Cells[3, I] := RowRecord.Subject;

RowCount := RowCount + 1;

Inc(I);

except

MessageBox(0,

PWideChar('Error! Error on step №' + IntToStr(I) +

'.'), 'Error', MB\_OK + MB\_ICONERROR);

end;

RowCount := RowCount - 2;

end;

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error',

MB\_OK + MB\_ICONERROR);

end;

end;

procedure TEditForm.Saveas1Click(Sender: TObject);

var

Put: file of TTimetable;

RowRecord: TTimetable;

I: Integer;

begin

if SaveTextFileDialog1.Execute then

try

AssignFile(Put, SaveTextFileDialog1.FileName);

Rewrite(Put);

with Timetable do

for I := 1 to RowCount - 1 do

begin

RowRecord.Professor := Cells[0, I];

RowRecord.Day := Cells[1, I];

RowRecord.Order := Cells[2, I];

RowRecord.Subject := Cells[3, I];

Write(Put, RowRecord);

end;

CloseFile(Put);

except

MessageBox(0, 'Error! Cannot assign file.', 'File error',

MB\_OK + MB\_ICONERROR);

end;

end;

procedure TEditForm.GridClear;

begin

Timetable.RowCount := 2;

Timetable.Cells[0, 1] := '';

Timetable.Cells[1, 1] := '';

Timetable.Cells[2, 1] := '';

Timetable.Cells[3, 1] := '';

end;

procedure TEditForm.FormCreate(Sender: TObject);

begin

with Timetable do

begin

Cells[0, 0] := 'Professor';

Cells[1, 0] := 'Day';

Cells[2, 0] := 'Order';

Cells[3, 0] := 'Subject';

ColWidths[0] := 250;

ColWidths[1] := 50;

ColWidths[2] := 50;

ColWidths[3] := 223;

Row := 1;

Col := 0;

end;

end;

end.

unit Unit6;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls, Vcl.Menus;

type

TModalView = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

ProfessorEdit: TEdit;

DayEdit: TEdit;

SubjectEdit: TEdit;

OrderEdit: TEdit;

CloseButton: TButton;

procedure CloseButtonClick(Sender: TObject);

private

{ Private declarations }

public

{ Public declarations }

end;

var

ModalView: TModalView;

implementation

{$R \*.dfm}

procedure TModalView.CloseButtonClick(Sender: TObject);

begin

ModalView.Close;

end;

end.

unit Unit7;

interface

uses

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

Vcl.Controls, Vcl.Forms, Vcl.Dialogs, Vcl.StdCtrls;

type

TModalEdit = class(TForm)

Label1: TLabel;

Label2: TLabel;

Label3: TLabel;

Label4: TLabel;

ProfessorEdit: TEdit;

SubjectEdit: TEdit;

CloseButton: TButton;

DeleteButton: TButton;

EditButton: TButton;

DayEdit: TComboBox;

OrderEdit: TComboBox;

procedure CloseButtonClick(Sender: TObject);

procedure EditButtonClick(Sender: TObject);

procedure DeleteButtonClick(Sender: TObject);

procedure DeleteRow(Current : Integer);

procedure RowSwop(First: Integer; Second: Integer);

procedure DayKeyPress(Sender: TObject; var key: Char);

procedure OrderKeyPress(Sender: TObject; var key: Char);

private

{ Private declarations }

public

{ Public declarations }

OpenedIndex : Integer;

end;

var

ModalEdit: TModalEdit;

implementation

{$R \*.dfm}

uses Unit5;

procedure TModalEdit.DayKeyPress(Sender: TObject; var key: Char);

begin

key := #0;

end;

procedure TModalEdit.OrderKeyPress(Sender: TObject; var key: Char);

begin

case key of

#8:

;

'1' .. '5':

if Length((Sender as TComboBox).Text + key) > 1 then

key := #0;

else

key := #0;

end;

end;

procedure TModalEdit.CloseButtonClick(Sender: TObject);

begin

ModalEdit.Close;

end;

procedure TModalEdit.DeleteButtonClick(Sender: TObject);

begin

DeleteRow(OpenedIndex);

end;

procedure TModalEdit.RowSwop(First: Integer; Second: Integer); { СОРТИРОВКА }

var

Temp: String;

I: Integer;

begin

with EditForm.Timetable do

for I := 0 to 3 do

begin

Temp := Cells[I, First];

Cells[I, First] := Cells[I, Second];

Cells[I, Second] := Temp;

end;

end;

procedure TModalEdit.DeleteRow(Current: Integer);

var

I : Integer;

begin

with EditForm.Timetable do

begin

for I := OpenedIndex to RowCount - 2 do

begin

RowSwop(I, I + 1);

end;

RowCount := RowCount - 1;

end;

ModalEdit.Close;

end;

procedure TModalEdit.EditButtonClick(Sender: TObject);

begin

with EditForm.Timetable do

begin

Cells[0, OpenedIndex] := ProfessorEdit.Text;

Cells[1, OpenedIndex] := DayEdit.Text;

Cells[2, OpenedIndex] := OrderEdit.Text;

Cells[3, OpenedIndex] := SubjectEdit.Text;

end;

end;

end.

**Код программы**

**(Java)**

import java.io.\*;

import java.util.ArrayList;

import java.util.Scanner;

class Record {

String professor, day, order, subject;

public Record(String readerStr) {

String[] reader = readerStr.split(" ", 4);

try {

this.professor = reader[0];

this.day = reader[1];

this.order = reader[2];

this.subject = reader[3];

} catch (Exception e) {

System.out.println("Incorrect data");

}

}

}

public class Main {

public static Scanner in = new Scanner(System.in);

public static ArrayList timetable = new ArrayList(0);

public static ArrayList<String> fileList = new ArrayList<String>(0);

public static void outToFile(ArrayList timetable) {

boolean inValid = true;

String fileName, timetableStr = printTimetableForFile(timetable);

FileWriter outF;

System.out.println("Enter name of output file:");

do {

try {

fileName = in.nextLine();

outF = new FileWriter("E:\\University\\OAiP\\LAB4\\Goal

1\\Java\\" + fileName + ".txt");

outF.write(timetableStr);

outF.close();

fileList.add(fileName);

inValid = false;

} catch (Exception e) {

System.out.println("Please try again");

}

} while (inValid);

}

public static String printTimetable(ArrayList timetable) {

String answer = "";

for (int i = 0; i < timetable.size(); i++) {

answer = answer + (i + 1) + " record | " + ((Record)

timetable.get(i)).professor + " | " + ((Record)

timetable.get(i)).day + " | " + ((Record)

timetable.get(i)).order + " | " + ((Record)

timetable.get(i)).subject + "\n";

}

return answer;

}

public static String printTimetableForFile(ArrayList timetable) {

String answer = "";

for (int i = 0; i < timetable.size(); i++) {

answer = answer + ((Record) timetable.get(i)).professor + " " +

((Record) timetable.get(i)).day + " " + ((Record)

timetable.get(i)).order + " " + ((Record)

timetable.get(i)).subject + "\n";

}

return answer;

}

private static void fileOpening() throws IOException {

FileReader inF = new FileReader("E:\\University\\OAiP\\LAB4\\Goal

1\\Java\\timetable\_1.txt");

Scanner fileScanner = new Scanner(inF);

String reader = "";

boolean inValid = true;

if (fileList.size() > 0) {

for (int i = 0; i < fileList.size(); i++) {

System.out.println("Choose file:\n" + (i + 1) + ". " +

fileList.get(i) + ".txt\n");

}

do {

try {

reader = in.nextLine();

if (Integer.parseInt(reader) < fileList.size() + 1) {

inF = new

FileReader("E:\\University\\OAiP\\LAB4\\Goal

1\\Java\\" +

fileList.get(Math.abs(Integer.parseInt(reader) –

1)) + ".txt");

fileScanner = new Scanner(inF);

inValid = false;

} else {

System.out.println("Try again");

}

} catch (Exception e) {

System.out.println("Try again");

}

} while (inValid);

try {

while (fileScanner.hasNextLine()) {

timetable.add(new Record(fileScanner.nextLine()));

}

} catch (Exception e) {

System.out.println("Try again");

}

}

fileScanner.close();

inF.close();

}

public static void createMain() { // СОЗДАНИЕ

boolean continueChoose = true;

String reader;

do {

System.out.println("Enter [professor day order subject]:");

reader = in.nextLine();

timetable.add(new Record(reader));

System.out.println("If you want to exit enter [exit] or press

[enter] to continue:");

if (in.nextLine().equals("exit")) {

continueChoose = false;

}

} while (continueChoose);

System.out.println("Timetable:\n" + printTimetable(timetable));

outToFile(timetable);

timetable.clear();

}

public static void viewMain() throws IOException { // ПРОСМОТР

fileOpening();

System.out.println("Timetable:\n" + printTimetable(timetable));

timetable.clear();

}

public static void editMain() throws IOException { // РЕДАКТИРОВАНИЕ

fileOpening();

System.out.println("Timetable:\n" + printTimetable(timetable));

boolean continueChoose = true;

String reader;

String i = "";

do {

System.out.println("Enter record you want to edit or any

another button + [enter] to continue:");

i = in.nextLine();

try {

if (Integer.parseInt(i) < timetable.size() + 1) {

System.out.println("Enter [professor day order subject]

or [delete] to remove record:");

reader = in.nextLine();

if (reader.equals("delete")) {

timetable.remove(Math.abs(Integer.parseInt(i) –

1));

} else {

timetable.set(Math.abs(Integer.parseInt(i) - 1),

new Record(reader));

}

}

System.out.println("Timetable:\n" +

printTimetable(timetable));

} catch (Exception e) {

System.out.println("Record not found");

}

System.out.println("If you want to exit enter [exit] or press

[enter] to continue:");

if (in.nextLine().equals("exit")) {

outToFile(timetable);

continueChoose = false;

}

} while (continueChoose);

timetable.clear();

}

public static void findMain() throws IOException { // ПОИСК

ArrayList timetableFound;

boolean inValid, continueChoose = true;

do {

timetableFound = new ArrayList(0);

inValid = true;

fileOpening();

System.out.println("Timetable:\n" + printTimetable(timetable));

System.out.println("Enter search criterion

(professor/day/order/subject):");

do {

switch (in.nextLine()) {

case "professor":

timetableFound = searchingByProf();

inValid = false;

break;

case "day":

timetableFound = searchingByDay();

inValid = false;

break;

case "order":

timetableFound = searchingByOrder();

inValid = false;

break;

case "subject":

timetableFound = searchingBySubject();

inValid = false;

break;

default:

System.out.println("Try again");

}

} while (inValid);

System.out.println("Found:\n" +

printTimetable(timetableFound));

System.out.println("Do you want save to file? [yes/...]");

if (in.nextLine().equals("yes")) {

outToFile(timetableFound);

}

System.out.println("If you want to exit enter [exit] or press

[enter] to continue:");

if (in.nextLine().equals("exit")) {

outToFile(timetable);

continueChoose = false;

}

timetable.clear();

} while (continueChoose);

}

private static ArrayList searchingByProf() {

ArrayList timetableFound = new ArrayList(0);

String search = "";

System.out.println("Enter search:");

search = in.nextLine();

for (int i = 0; i < timetable.size(); i++) {

if (((Record) timetable.get(i)).professor.equals(search)) {

timetableFound.add(timetable.get(i));

}

}

return timetableFound;

}

private static ArrayList searchingByDay() {

ArrayList timetableFound = new ArrayList(0);

String search = "";

System.out.println("Enter search:");

search = in.nextLine();

for (int i = 0; i < timetable.size(); i++) {

if (((Record) timetable.get(i)).day.equals(search)) {

timetableFound.add(timetable.get(i));

}

}

return timetableFound;

}

private static ArrayList searchingByOrder() {

ArrayList timetableFound = new ArrayList(0);

String search = "";

System.out.println("Enter search:");

search = in.nextLine();

for (int i = 0; i < timetable.size(); i++) {

if (((Record) timetable.get(i)).order.equals(search)) {

timetableFound.add(timetable.get(i));

}

}

return timetableFound;

}

private static ArrayList searchingBySubject() {

ArrayList timetableFound = new ArrayList(0);

String search = "";

System.out.println("Enter search:");

search = in.nextLine();

for (int i = 0; i < timetable.size(); i++) {

if (((Record) timetable.get(i)).subject.equals(search)) {

timetableFound.add(timetable.get(i));

}

}

return timetableFound;

}

public static boolean hubChoosing() throws IOException {

boolean inValid = true;

System.out.println("Choose action:\n1. Create\n2. View\n3. Find\n4.

Edit\n5. Exit");

do {

switch (in.nextLine()) {

case "1":

createMain();

inValid = false;

break;

case "2":

viewMain();

inValid = false;

break;

case "3":

findMain();

inValid = false;

break;

case "4":

editMain();

inValid = false;

break;

case "5":

return false;

default:

System.out.println("Please try again.");

}

} while (inValid);

return true;

}

public static void main(String[] args) throws IOException {

boolean continueChoose = true;

System.out.println("This program imitate work with timetable.");

while (continueChoose) {

continueChoose = hubChoosing();

}

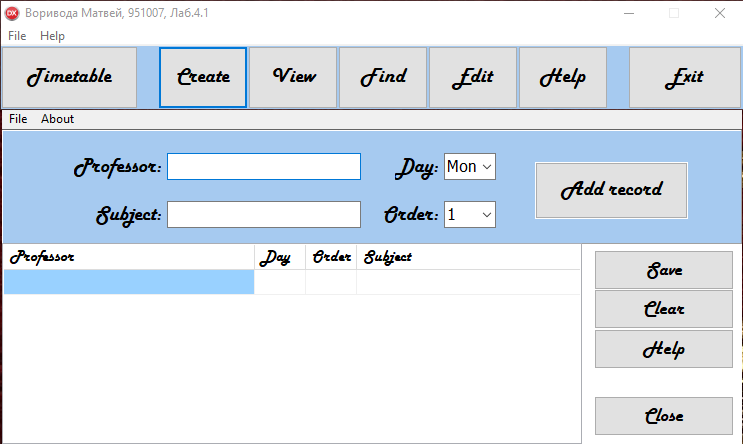
System.out.println("Goodbye!");

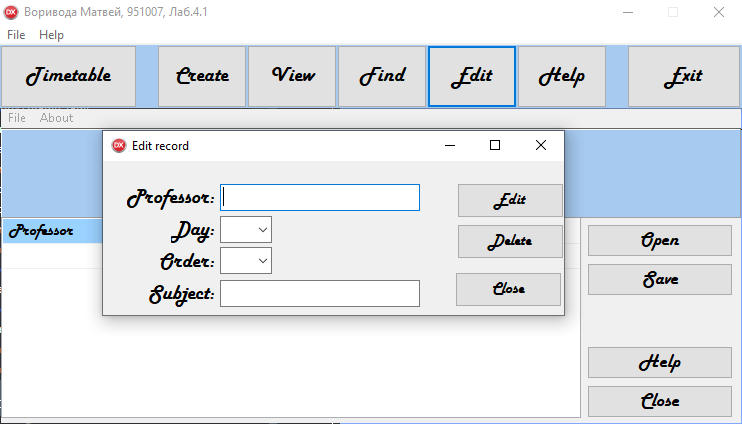
}

}

**Скриншоты**

А) Delphi





Б) Java

