УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №4.1

по предмету

Основы алгоритмизации и программирования

Вариант 3

Выполнил:

Бетеня К.С.

Проверила:

Данилова Г.В.

Группа 351005

Минск 2024

Задание:

1. Сведения о футбольных командах, участвующих в чемпионате мира

* страна;
* название команды;
* фамилия главного тренера;
* рейтинг команды.

Вывести сведения о футбольных командах, участвующих в чемпионате мира в порядке убывания их рейтинга.

Код программы на **Delphi**:

**Unit MainFormUnit;**

Interface

Uses

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Variants,

System.Classes,

Vcl.Graphics,

Vcl.Controls,

Vcl.Forms,

Vcl.Dialogs,

Vcl.Buttons,

Vcl.ExtCtrls,

Vcl.StdCtrls,

Vcl.Menus,

Vcl.Grids,

System.ImageList,

Vcl.ImgList,

Vcl.Imaging.Pngimage;

Type

TMainForm = Class(TForm)

TMLabel: TLabel;

NavPanel: TPanel;

AddSpButton: TSpeedButton;

ChangeSpButton: TSpeedButton;

SearchSpButton: TSpeedButton;

RemoveSpButton: TSpeedButton;

ExitSpButton: TSpeedButton;

NavImgList: TImageList;

MainMenu: TMainMenu;

FileButton: TMenuItem;

InstractionButton: TMenuItem;

AboutEditorButton: TMenuItem;

CupImage: TImage;

AboutCupLabel: TLabel;

PointTabelStrGrid: TStringGrid;

BestFormLabel: TLabel;

ExposureLabel: TLabel;

StGridInfo: TLabel;

CloseButton: TMenuItem;

MMIconImgList: TImageList;

Procedure FormCloseQuery(Sender: TObject; Var CanClose: Boolean);

Procedure CloseButtonClick(Sender: TObject);

Procedure ExitSpButtonClick(Sender: TObject);

Procedure FormCreate(Sender: TObject);

Procedure AboutEditorButtonClick(Sender: TObject);

Procedure InstractionButtonClick(Sender: TObject);

Procedure AddSpButtonClick(Sender: TObject);

Procedure ChangeSpButtonClick(Sender: TObject);

Procedure RemoveSpButtonClick(Sender: TObject);

Procedure SearchSpButtonClick(Sender: TObject);

Procedure PointTabelStrGridKeyUp(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure PointTabelStrGridKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure PointTabelStrGridDblClick(Sender: TObject);

Procedure FormKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Private

Procedure WMGetMinMaxInfo(Var Msg: TWMGetMinMaxInfo);

Public

{ Public declarations }

End;

Var

MainForm: TMainForm;

CurentRow: Integer;

Implementation

{$R \*.dfm}

Uses

FrontendUnit,

AddRecordUnit,

BackendUnit,

ChangeRecordUnit,

SearchRecordUnit;

Procedure TMainForm.AboutEditorButtonClick(Sender: TObject);

Begin

CreateModalForm('О разработчике', 'Выполнил студент группы 351005 Бетеня

Константин.', Screen.Width \* 21 Div 100,

Screen.Height \* 7 Div 100);

End;

Procedure TMainForm.AddSpButtonClick(Sender: TObject);

Begin

If (CurentRecordsCount = TEAMS\_COUNT) Then

MessageBox(0, 'Вы достигли максимального колличества команд!', 'Ошибка',

MB\_ICONERROR)

Else

Begin

Application.CreateForm(TAddRecordForm, AddRecordForm);

AddRecordForm.ShowModal;

End;

End;

Procedure TMainForm.ChangeSpButtonClick(Sender: TObject);

Var

TempRecord: TFootStatsRecord;

Begin

CurentRow := MainForm.PointTabelStrGrid.Row;

If (CurentRow <> 0) Then

Begin

Application.CreateForm(TChangeRecordForm, ChangeRecordForm);

TempRecord := GetRecordFromFile(CurentRow);

ChangeRecordForm.CountryLabeledEdit.Text := TempRecord.Country;

ChangeRecordForm.TeamNameLabeledEdit.Text := TempRecord.Team;

ChangeRecordForm.CoachLabeledEdit.Text := TempRecord.Coach;

ChangeRecordForm.PointsLabeledEdit.Text := IntToStr(TempRecord.Points);

ChangeRecordForm.Showmodal;

End

Else

MessageBox(0, 'Выберите строку записи!', 'Ошибка', MB\_ICONERROR);

End;

Procedure TMainForm.CloseButtonClick(Sender: TObject);

Begin

MainForm.Close;

End;

Procedure TMainForm.ExitSpButtonClick(Sender: TObject);

Begin

MainForm.Close;

End;

Procedure TMainForm.FormCloseQuery(Sender: TObject; Var CanClose: Boolean);

Var

ResultKey: Integer;

Begin

ResultKey := Application.Messagebox('Вы уверены, что хотите закрыть оконное

приложение?', 'Выход', MB\_YESNO +

MB\_ICONQUESTION + MB\_DEFBUTTON2);

If ResultKey = ID\_YES Then

Begin

CanClose := True;

LoadRecordsInFile;

End

Else

CanClose := False;

End;

Procedure TMainForm.FormCreate(Sender: TObject);

Begin

PointTabelStrGrid.ColWidths[0] := (PointTabelStrGrid.Width \* 9) Div 100;

PointTabelStrGrid.ColWidths[1] := (PointTabelStrGrid.Width \* 23) Div 100;

PointTabelStrGrid.ColWidths[2] := (PointTabelStrGrid.Width \* 23) Div 100;

PointTabelStrGrid.ColWidths[3] := (PointTabelStrGrid.Width \* 23) Div 100;

PointTabelStrGrid.ColWidths[4] := (PointTabelStrGrid.Width \* 21) Div 100;

PointTabelStrGrid.Cells[0, 0] := 'Место';

PointTabelStrGrid.Cells[1, 0] := 'Страна';

PointTabelStrGrid.Cells[2, 0] := 'Название команды';

PointTabelStrGrid.Cells[3, 0] := 'Главный тренер';

PointTabelStrGrid.Cells[4, 0] := 'Итоговый рейтинг';

LoadRecordsFromFile();

If (CurentRecordsCount <> 0) Then

PointTabelStrGrid.FixedRows := 1;

End;

Function TMainForm.FormHelp(Command: Word; Data: NativeInt; Var CallHelp:

Boolean): Boolean;

Begin

CallHelp := False;

End;

Procedure TMainForm.FormKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_INSERT) Then

AddSPButton.Click;

If (Key = VK\_ESCAPE) Then

MainForm.Close;

End;

Procedure TMainForm.InstractionButtonClick(Sender: TObject);

Var

InstractionText: String;

Begin

InstractionText := 'Инструкция:' +

#13#10 + '1. Чтобы добавить новую запись нажмите на кнопку ''' +

AddSpButton.Caption + ''';' +

#13#10 + '2. Если вы хотите изменить какую-то запись, то

выберите необходимую' +

#13#10 + ' строку для изменения и нажимайте на кнопку ''' +

ChangeSpButton.Caption + ''';' +

#13#10 + '3. Если у вас много записей и вы потерялись, то нажмите на кнопку' +

#13#10 + ' ''' + SearchSpButton.Caption + '''. Там вы можете выбрать

удобный вам критерий для поиска' +

#13#10 + ' и даже сразу изменить эту запись;' +

#13#10 + '4. Если запись необходимо убрать из турнирной таблицы, то нажмите' +

#13#10 + ' на кнопку ''' + RemoveSpButton.Caption + '''. Для этого так же

необходимо заранее выбрать' +

#13#10 + ' нужную строку.' + #13#10 +

#13#10 + 'Если вы хотите выйти, то у вас есть много вариантов, как это

сделать:' +

#13#10 + ' - Сочетание клавишь Alt + F4;' +

#13#10 + ' - Сочетание клавишь Ctrl + Q;' +

#13#10 + ' - Нажать ESCape (работает, как способ закрытия любого окна);' +

#13#10 + ' - Выбрать кнопку ''' + ExitSpButton.Caption + '''

в левом меню;' +

#13#10 + ' - Во вкладке ' + FileButton.Caption + ' пункт ''' +

CloseButton.Caption + '''.' +

#13#10 + #13#10 + 'Работа с файлами: ' +

#13#10 + ' В форме автоматически реализованы типизированный файл и файл' +

#13#10 + ' корректур. Если вы сделали запись и закрыли форму, то всё' +

#13#10 + ' сохранится автоматически и при запуске будет выгружено. ' +

#13#10 + #13#10 + 'В любой форме вы можете обратится к справке нажав клавишу

F1 или' + #13#10 + 'выбрав её в верхем меню.' + #13#10 +

#13#10 + 'В турнирной таблице записи упорядочены по убыванию рейтинга

команд.' + #13#10 + #13#10 + 'В случае когда набрано одинаковое кол-

во очков, первой будет идти' +

#13#10 + 'команда, которая была записана в таблицу раньше.' +

#13#10 + 'Горячие клавиши:' +

#13#10 + ' - Двойное нажатие вызовет изменения выбранной записи;' +

#13#10 + ' - Нажатие Delete вызовет удаление выбранной записи;' +

#13#10 + ' - Нажатие Insert вызовет добавление новой записи;' +

#13#10 + ' - Нажатие ESCape закрывает текущую форму.';

CreateModalForm('Инструкция', InstractionText, Screen.Width \* 29 Div 100,

Screen.Height \* 70 Div 100);

End;

Procedure TMainForm.PointTabelStrGridDblClick(Sender: TObject);

Begin

ChangeSPButton.Click;

End;

Procedure TMainForm.PointTabelStrGridKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_INSERT) Then

AddSPButton.Click;

If (Key = VK\_DELETE) Then

RemoveSPButton.Click;

If (Key = VK\_ESCAPE) Then

MainForm.Close;

End;

Procedure TMainForm.PointTabelStrGridKeyUp(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (CurentRecordsCount <> 0) Then

PointTabelStrGrid.FixedRows := 1

Else

PointTabelStrGrid.FixedRows := 0;

End;

Procedure TMainForm.RemoveSpButtonClick(Sender: TObject);

Var

ResultKey: Integer;

Begin

CurentRow := MainForm.PointTabelStrGrid.Row;

If (CurentRow <> 0) Then

Begin

ResultKey := Application.Messagebox('Вы уверены, что хотите удалить

запись?', 'Удаление записи',

MB\_YESNO + MB\_ICONQUESTION + MB\_DEFBUTTON2);

If ResultKey = ID\_YES Then

Begin

DeleteRow(CurentRow);

Dec(CurentRecordsCount);

If CurentRecordsCount > 11 Then

PointTabelStrGrid.ColWidths[4] :=

(PointTabelStrGrid.Width \* 19) Div 100

Else

MainForm.PointTabelStrGrid.ColWidths[4] :=

(MainForm.PointTabelStrGrid.Width \* 21) Div 100;

SortRecords();

InputRecordsInTableGrid();

End;

End

Else

MessageBox(0, 'Необходимо выбрать строку для удаления!', 'Ошибка',

MB\_ICONERROR);

End;

Procedure TMainForm.SearchSpButtonClick(Sender: TObject);

Begin

If (CurentRecordsCount <> 0) Then

Begin

Application.CreateForm(TSearchRecordForm, SearchRecordForm);

SearchRecordForm.ShowModal;

End

Else

MessageBox(0, 'Для поиска необходимо создать хотя-бы одну запись!',

'Ошибка', MB\_ICONERROR);

End;

Procedure TMainForm.WMGetMinMaxInfo(Var Msg: TWMGetMinMaxInfo);

Begin

Msg.MinMaxInfo.PtMaxSize.X := Width;

Msg.MinMaxInfo.PtMaxSize.Y := Height;

Msg.MinMaxInfo.PtMaxTrackSize.X := Width;

Msg.MinMaxInfo.PtMaxTrackSize.Y := Height;

Left := (Screen.Width - Width) Div 2;

Top := (Screen.Height - Height) Div 2;

End;

End.

**Unit LoadingScreenUnit;**

Interface

Uses

Winapi.Windows,

Vcl.Forms,

Vcl.ExtCtrls,

Vcl.Imaging.Jpeg,

System.Classes,

Vcl.Controls;

Type

TLoadingScreen = Class(TForm)

EndLoadingScreen: TTimer;

AlphaBlendChanging: TTimer;

Image1: TImage;

Procedure EndLoadingScreenTimer(Sender: TObject);

Procedure FormCreate(Sender: TObject);

Procedure AlphaBlendChangingTimer(Sender: TObject);

End;

Var

LoadingScreen: TLoadingScreen;

Implementation

{$R \*.dfm}

Uses

MainFormUnit;

Procedure TLoadingScreen.FormCreate(Sender: TObject);

Var

HRgn: Cardinal;

Begin

HRgn := CreateEllipticRgn(0, 0, 500, 500);

SetWindowRgn(Handle, HRgn, False);

EndLoadingScreen.Enabled := True;

AlphaBlendChanging.Enabled := True;

Left := (Screen.Width - Width) Div 2;

Top := (Screen.Height - Height) Div 2;

End;

Procedure TLoadingScreen.AlphaBlendChangingTimer(Sender: TObject);

Begin

If LoadingScreen.AlphaBlendValue > 253 Then

AlphaBlendChanging.Enabled := False

Else

LoadingScreen.AlphaBlendValue := LoadingScreen.AlphaBlendValue + 2;

End;

Procedure TLoadingScreen.EndLoadingScreenTimer(Sender: TObject);

Begin

EndLoadingScreen.Enabled := False;

End;

End.

**Unit ChangeRecordUnit;**

Interface

Uses

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Classes,

Vcl.Forms,

Vcl.Menus,

Vcl.StdCtrls,

Vcl.ExtCtrls,

Clipbrd,

Vcl.Mask,

Vcl.Controls;

Type

TLabeledEdit = Class(Vcl.ExtCtrls.TLabeledEdit)

Public

Procedure WMPaste(Var Msg: TMessage); Message WM\_PASTE;

End;

TChangeRecordForm = Class(TForm)

MainMenu: TMainMenu;

RefferenceButton: TMenuItem;

CountryLabel: TLabel;

TeamNameLabel: TLabel;

CoachLabel: TLabel;

PointsLabel: TLabel;

CountryLabeledEdit: TLabeledEdit;

TeamNameLabeledEdit: TLabeledEdit;

CoachLabeledEdit: TLabeledEdit;

PointsLabeledEdit: TLabeledEdit;

ChangeButton: TButton;

Procedure CountryLabeledEditChange(Sender: TObject);

Procedure TeamNameLabeledEditChange(Sender: TObject);

Procedure CoachLabeledEditChange(Sender: TObject);

Procedure PointsLabeledEditChange(Sender: TObject);

Procedure CountryLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled: Boolean);

Procedure TeamNameLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled: Boolean);

Procedure PointsLabeledEditContextPopup(Sender: TObject; MousePos: TPoint;

Var Handled: Boolean);

Procedure CoachLabeledEditContextPopup(Sender: TObject; MousePos: TPoint;

Var Handled: Boolean);

Procedure CountryLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Procedure TeamNameLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Procedure CoachLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Procedure PointsLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Procedure PointsLabeledEditKeyPress(Sender: TObject; Var Key: Char);

Procedure FormCreate(Sender: TObject);

Procedure ChangeButtonClick(Sender: TObject);

Procedure RefferenceButtonClick(Sender: TObject);

Procedure FormKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Private

{ Private declarations }

Public

{ Public declarations }

End;

Var

ChangeRecordForm: TChangeRecordForm;

Implementation

{$R \*.dfm}

Uses

FrontendUnit,

BackendUnit,

MainFormUnit;

Procedure TChangeRecordForm.ChangeButtonClick(Sender: TObject);

Var

ResultKey: Integer;

Begin

If Not IfRecordExist(CountryLabeledEdit.Text, CoachLabeledEdit.Text,

TeamNameLabeledEdit.Text, PointsLabeledEdit.Text) Then

ResultKey := ID\_YES

Else

ResultKey := Application.Messagebox

('Подобная запись уже существует в таблице результатов, вы уверенны,

что хотите создать дубликат?', 'Выход', MB\_YESNO + MB\_ICONQUESTION +

MB\_DEFBUTTON2);

If (ResultKey = ID\_YES) Then

Begin

ChangeRecordInFile(StrToWideChar(CountryLabeledEdit.Text),

StrToWideChar(TeamNameLabeledEdit.Text),

StrToWideChar(CoachLabeledEdit.Text),

StrToint(PointsLabeledEdit.Text), CurentRow);

SortRecords();

InputRecordsInTableGrid();

ChangeRecordForm.Close;

End;

End;

Procedure TChangeRecordForm.CoachLabeledEditChange(Sender: TObject);

Begin

ChangeButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit,

PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TChangeRecordForm.CoachLabeledEditContextPopup(Sender: TObject;

MousePos: TPoint; Var

Handled: Boolean);

Begin

Handled := True;

End;

Procedure TChangeRecordForm.CoachLabeledEditKeyDown(Sender: TObject; Var Key:

Word; Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

ChangeRecordForm.Close;

End;

Procedure TChangeRecordForm.CountryLabeledEditChange(Sender: TObject);

Begin

ChangeButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit,

PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TChangeRecordForm.CountryLabeledEditContextPopup(Sender: TObject;

MousePos: TPoint; Var

Handled: Boolean);

Begin

Handled := True;

End;

Procedure TChangeRecordForm.CountryLabeledEditKeyDown(Sender: TObject; Var Key:

Word; Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

ChangeRecordForm.Close;

End;

Procedure TChangeRecordForm.FormCreate(Sender: TObject);

Begin

CountryLabeledEdit.EditLabel.Caption := '';

TeamNameLabeledEdit.EditLabel.Caption := '';

CoachLabeledEdit.EditLabel.Caption := '';

PointsLabeledEdit.EditLabel.Caption := '';

End;

Procedure TChangeRecordForm.FormKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Begin

If (Key = VK\_ESCAPE) Then

ChangeRecordForm.Close;

End;

Procedure TChangeRecordForm.PointsLabeledEditChange(Sender: TObject);

Begin

ChangeButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit,

PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TChangeRecordForm.PointsLabeledEditContextPopup(Sender: TObject;

MousePos: TPoint; Var

Handled: Boolean);

Begin

Handled := True;

End;

Procedure TChangeRecordForm.PointsLabeledEditKeyDown(Sender: TObject; Var Key:

Word; Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

ChangeRecordForm.Close;

End;

Procedure TChangeRecordForm.PointsLabeledEditKeyPress(Sender: TObject;

Var Key: Char);

Begin

Key := CheckInput(Key, PointsLabeledEdit);

End;

Procedure TChangeRecordForm.RefferenceButtonClick(Sender: TObject);

Var

InstractionTest: String;

Begin

InstractionTest := ' - В первых трёх полях ограничений на ввод нет за

исключением' + #13#10 + 'того, что максимальная длина

записи 20 символов. Опасайтесь' + #13#10 + 'пробелов, они

не видны в турнирной таблице.' + #13#10 + ' - У поля

итогового рейтинга команды есть ограничения: ' + #13#10 +

'Там можно писать только натуральные числа в диапазоне

от 0' + #13#10 + 'до 100.' + #13#10 + ' - После того как

все поля изменены нажимайте на кнопку' + #13#10 + '''' +

ChangeButton.Caption + ''';';

CreateModalForm('Справка', InstractionTest, Screen.Width \* 27 Div 100,

Screen.Height \* 21 Div 100);

End;

Procedure TChangeRecordForm.TeamNameLabeledEditChange(Sender: TObject);

Begin

ChangeButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit,

PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TChangeRecordForm.TeamNameLabeledEditContextPopup(Sender: TObject; MousePos: TPoint; Var Handled: Boolean);

Begin

Handled := True;

End;

Procedure TChangeRecordForm.TeamNameLabeledEditKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

ChangeRecordForm.Close;

End;

Procedure TLabeledEdit.WMPaste(Var Msg: TMessage);

Begin

If Clipboard.HasFormat(CF\_TEXT) Then

Begin

Try

If (ChangeRecordForm.ActiveControl =

ChangeRecordForm.PointsLabeledEdit) And

Not IsCorrectPointsClipboard(Clipboard.AsText,

ChangeRecordForm.PointsLabeledEdit)

Then

Raise Exception.Create('Некорректное количество

очков!'#13#10'Максимальное кол-во очков

100 :(');

If (ChangeRecordForm.ActiveControl =

ChangeRecordForm.CoachLabeledEdit) And

Not IsCorrectStrings(Clipboard.AsText,

ChangeRecordForm.CoachLabeledEdit) Then

Raise Exception.Create('Некорректная фамилия главного

тренера!'#13#10'Давайте уместимся в 20

символов.');

If (ChangeRecordForm.ActiveControl =

ChangeRecordForm.TeamNameLabeledEdit) And

Not IsCorrectStrings(Clipboard.AsText,

ChangeRecordForm.TeamNameLabeledEdit) Then

Raise Exception.Create('Некорректное название

команды!'#13#10'Нужно более корокое

название.');

If (ChangeRecordForm.ActiveControl =

ChangeRecordForm.CountryLabeledEdit) And

Not IsCorrectStrings(Clipboard.AsText,

ChangeRecordForm.CountryLabeledEdit) Then

Raise Exception.Create('Слишком длинное название

страны!'#13#10'Используйте сокращения

:)');

Except

On E: Exception Do

Begin

MessageBox(0, PWideChar(E.Message), 'Ошибка', MB\_ICONERROR);

Exit;

End;

End;

End;

Inherited;

End;

End.

**Unit SearchRecordUnit;**

Interface

Uses

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Classes,

Vcl.Forms,

Vcl.Menus,

Vcl.StdCtrls,

Vcl.ExtCtrls,

Clipbrd,

Vcl.Mask,

Vcl.Controls;

Type

TLabeledEdit = Class(Vcl.ExtCtrls.TLabeledEdit)

Public

Procedure WMPaste(Var Msg: TMessage); Message WM\_PASTE;

End;

TItems = Array [1 .. 4] Of String;

TSearchRecordForm = Class(TForm)

MainMenu: TMainMenu;

ReferenceButton: TMenuItem;

CBox: TComboBox;

CBoxLabel: TLabel;

SearchStrLabel: TLabel;

SearchStrLEdit: TLabeledEdit;

SearchButton: TButton;

ResultLabel: TLabel;

ChangeRecordButton: TButton;

Procedure ReferenceButtonClick(Sender: TObject);

Procedure FormCreate(Sender: TObject);

Procedure CBoxCloseUp(Sender: TObject);

Procedure SearchStrLEditKeyPress(Sender: TObject; Var Key: Char);

Procedure SearchStrLEditKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure SearchStrLEditChange(Sender: TObject);

Procedure SearchButtonClick(Sender: TObject);

Procedure ChangeRecordButtonClick(Sender: TObject);

Procedure FormKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Procedure CBoxKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Private

{ Private declarations }

Public

{ Public declarations }

End;

Var

SearchRecordForm: TSearchRecordForm;

StrIndex: Integer;

Items: TItems = ('по стране', 'по названию команды', 'по фамилии гл. тренера',

'по итоговому рейтингу');

Implementation

{$R \*.dfm}

Uses

MainFormUnit,

FrontendUnit,

BackendUnit,

ChangeRecordUnit;

Procedure TSearchRecordForm.CBoxKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Begin

If (Key = VK\_ESCAPE) Then

SearchRecordForm.Close;

End;

Procedure TSearchRecordForm.ChangeRecordButtonClick(Sender: TObject);

Var

TempRecord: TFootStatsRecord;

Begin

Application.CreateForm(TChangeRecordForm, ChangeRecordForm);

CurentRow := StrIndex + 1;

TempRecord := GetRecordFromFile(CurentRow);

ChangeRecordForm.CountryLabeledEdit.Text := TempRecord.Country;

ChangeRecordForm.TeamNameLabeledEdit.Text := TempRecord.Team;

ChangeRecordForm.CoachLabeledEdit.Text := TempRecord.Coach;

ChangeRecordForm.PointsLabeledEdit.Text := IntToStr(TempRecord.Points);

ChangeRecordForm.Showmodal;

SearchRecordForm.Close;

End;

Procedure TSearchRecordForm.CBoxCloseUp(Sender: TObject);

Begin

SearchStrLEdit.Text := '';

SearchButton.Enabled := False;

ResultLabel.Caption := '';

ChangeRecordButton.Visible := False;

If (CBox.ItemIndex In [0 .. 2]) Then

Begin

SearchStrLEdit.MaxLength := 20;

SearchStrLEdit.TextHint := 'строка для поиска';

End

Else

Begin

SearchStrLEdit.MaxLength := 3;

SearchStrLEdit.TextHint := '0..100';

End;

End;

Procedure TSearchRecordForm.FormCreate(Sender: TObject);

Begin

CBox.Items.AddStrings(Items);

SearchStrLEdit.EditLabel.Caption := '';

End;

Procedure TSearchRecordForm.FormKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_ESCAPE) Then

SearchRecordForm.Close;

End;

Procedure TSearchRecordForm.ReferenceButtonClick(Sender: TObject);

Var

InstractionTest: String;

Begin

InstractionTest := ' - В первых трёх вариантах поиска ограничений на ввод нет

за' + #13#10 + 'исключением того, что максимальная длина

записи 20 символов.' + #13#10 + ' - У поля итогового

рейтинга команды есть ограничения: ' + #13#10 + 'Там можно

писать только натуральные числа в диапазоне от 0' + #13#10

+ 'до 100.' + #13#10 + ' - После того как все поля

заполнены нажимайте на кнопку' + #13#10 + '''' +

SearchButton.Caption + ''';' + #13#10 + #13#10 +

'Если вы хотите изменить данные этой строки, то нажмите' +

#13#10 + 'на кнопку ''' + ChangeRecordButton.Caption +

'''.';

CreateModalForm('Справка', InstractionTest, Screen.Width \* 27 Div 100,

Screen.Height \* 24 Div 100);

End;

Procedure TSearchRecordForm.SearchButtonClick(Sender: TObject);

Begin

StrIndex := IndexRecord(CBox.ItemIndex, StrToWideChar(SearchStrLEdit.Text));

If StrIndex <> -1 Then

Begin

ResultLabel.Caption := CreateResultGrid(StrIndex + 1);

ChangeRecordButton.Visible := True;

End

Else

MessageBox(0, 'Запись по вашему параметру не найдена!', 'Ошибка',

MB\_ICONERROR);

End;

Procedure TSearchRecordForm.SearchStrLEditChange(Sender: TObject);

Begin

ResultLabel.Caption := '';

ChangeRecordButton.Visible := False;

SearchButton.Enabled := SearchStrLEdit.Text <> '';

End;

Procedure TSearchRecordForm.SearchStrLEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

SearchRecordForm.Close;

End;

Procedure TSearchRecordForm.SearchStrLEditKeyPress(Sender: TObject; Var

Key: Char);

Begin

If CBox.Text = '' Then

Key := #0;

If CBox.ItemIndex = 4 Then

Key := CheckInput(Key, SearchStrLEdit);

End;

Procedure TLabeledEdit.WMPaste(Var Msg: TMessage);

Begin

If Clipboard.HasFormat(CF\_TEXT) Then

Begin

Try

If (SearchRecordForm.CBox.ItemIndex = 3) And Not

IsCorrectPointsClipboard(Clipboard.AsText,

SearchRecordForm.SearchStrLEdit) Then

Raise Exception.Create('Некорректное количество

очков!'#13#10'Максимальное кол-во очков

100 :(');

If (SearchRecordForm.CBox.ItemIndex = 2) And Not

IsCorrectStrings(Clipboard.AsText,

SearchRecordForm.SearchStrLEdit) Then

Raise Exception.Create('Некорректная фамилия главного

тренера!'#13#10'Давайте уместимся в 20

символов.');

If (SearchRecordForm.CBox.ItemIndex = 1) And Not

IsCorrectStrings(Clipboard.AsText,

SearchRecordForm.SearchStrLEdit) Then

Raise Exception.Create('Некорректное название

команды!'#13#10'Нужно более корокое

название.');

If (SearchRecordForm.CBox.ItemIndex = 0) And Not

IsCorrectStrings(Clipboard.AsText,

SearchRecordForm.SearchStrLEdit) Then

Raise Exception.Create('Слишком длинное название

страны!'#13#10'Используйте сокращения

:)');

Except

On E: Exception Do

Begin

MessageBox(0, PWideChar(E.Message), 'Ошибка', MB\_ICONERROR);

Exit;

End;

End;

End;

Inherited;

End;

End.

**Unit AddRecordUnit;**

Interface

Uses

Clipbrd,

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Classes,

Vcl.Forms,

Vcl.Menus,

Vcl.StdCtrls,

Vcl.ExtCtrls,

Vcl.Controls,

Vcl.Mask;

Type

TLabeledEdit = Class(Vcl.ExtCtrls.TLabeledEdit)

Public

Procedure WMPaste(Var Msg: TMessage); Message WM\_PASTE;

End;

TAddRecordForm = Class(TForm)

MainMenu: TMainMenu;

ReferenceButton: TMenuItem;

CountryLabel: TLabel;

TeamNameLabel: TLabel;

CoachLabel: TLabel;

PointsLabel: TLabel;

CountryLabeledEdit: TLabeledEdit;

TeamNameLabeledEdit: TLabeledEdit;

CoachLabeledEdit: TLabeledEdit;

PointsLabeledEdit: TLabeledEdit;

AddButton: TButton;

Procedure FormCreate(Sender: TObject);

Procedure ReferenceButtonClick(Sender: TObject);

Procedure CountryLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled: Boolean);

Procedure CountryLabeledEditKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure CoachLabeledEditKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure TeamNameLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Procedure PointsLabeledEditKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Procedure TeamNameLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled: Boolean);

Procedure CoachLabeledEditContextPopup(Sender: TObject; MousePos: TPoint;

Var Handled: Boolean);

Procedure PointsLabeledEditContextPopup(Sender: TObject; MousePos: TPoint;

Var Handled: Boolean);

Procedure PointsLabeledEditKeyPress(Sender: TObject; Var Key: Char);

Procedure CountryLabeledEditChange(Sender: TObject);

Procedure PointsLabeledEditChange(Sender: TObject);

Procedure CoachLabeledEditChange(Sender: TObject);

Procedure TeamNameLabeledEditChange(Sender: TObject);

Procedure AddButtonClick(Sender: TObject);

Procedure FormKeyDown(Sender: TObject; Var Key: Word; Shift: TShiftState);

Private

{ Private declarations }

Public

{ Public declarations }

End;

Const

MAX\_STR\_LENGTH: Integer = 20;

MIN\_POINTS: Integer = 0;

MAX\_POINTS: Integer = 100;

Var

AddRecordForm: TAddRecordForm;

Implementation

{$R \*.dfm}

Uses

FrontendUnit,

MainFormUnit,

BackendUnit;

Procedure TAddRecordForm.AddButtonClick(Sender: TObject);

Var

ResultKey: Integer;

Begin

If Not IfRecordExist(CountryLabeledEdit.Text, CoachLabeledEdit.Text,

TeamNameLabeledEdit.Text, PointsLabeledEdit.Text) Then

ResultKey := ID\_YES

Else

ResultKey := Application.Messagebox

('Подобная запись уже существует в таблице результатов, вы уверенны,

что хотите создать дубликат?', 'Выход', MB\_YESNO + MB\_ICONQUESTION +

MB\_DEFBUTTON2);

If (ResultKey = ID\_YES) Then

Begin

InputRecordInFile(StrToWideChar(CountryLabeledEdit.Text),

StrToWideChar(TeamNameLabeledEdit.Text),

StrToWideChar(CoachLabeledEdit.Text),

StrToint(PointsLabeledEdit.Text));

Inc(CurentRecordsCount);

SortRecords();

InputRecordsInTableGrid();

MainForm.PointTabelStrGrid.FixedRows := 1;

CountryLabeledEdit.Text := '';

TeamNameLabeledEdit.Text := '';

CoachLabeledEdit.Text := '';

PointsLabeledEdit.Text := '';

AddRecordForm.Close;

End;

End;

Procedure TAddRecordForm.CoachLabeledEditChange(Sender: TObject);

Begin

AddButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit, PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TAddRecordForm.CoachLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled:

Boolean);

Begin

Handled := True;

End;

Procedure TAddRecordForm.CoachLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

AddRecordForm.Close;

End;

Procedure TAddRecordForm.CountryLabeledEditChange(Sender: TObject);

Begin

AddButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit, CoachLabeledEdit, PointsLabeledEdit, TeamNameLabeledEdit);

End;

Procedure TAddRecordForm.CountryLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled:

Boolean);

Begin

Handled := True;

End;

Procedure TAddRecordForm.CountryLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

AddRecordForm.Close;

End;

Procedure TAddRecordForm.FormCreate(Sender: TObject);

Begin

CountryLabeledEdit.EditLabel.Caption := '';

TeamNameLabeledEdit.EditLabel.Caption := '';

CoachLabeledEdit.EditLabel.Caption := '';

PointsLabeledEdit.EditLabel.Caption := '';

End;

Procedure TAddRecordForm.FormKeyDown(Sender: TObject; Var Key: Word; Shift:

TShiftState);

Begin

If (Key = VK\_ESCAPE) Then

AddRecordForm.Close;

End;

Procedure TAddRecordForm.PointsLabeledEditChange(Sender: TObject);

Begin

AddButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit, PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TAddRecordForm.PointsLabeledEditContextPopup(Sender: TObject; MousePos:

TPoint; Var Handled:

Boolean);

Begin

Handled := True;

End;

Procedure TAddRecordForm.PointsLabeledEditKeyDown(Sender: TObject; Var Key: Word;

Shift: TShiftState);

Var

CurKey: Char;

Begin

If Not Clipboard.HasFormat(CF\_TEXT) And (Key = VK\_INSERT) Then

Key := 0;

If (Key = VK\_DELETE) And (PointsLabeledEdit.SelText = '') Then

Begin

CurKey := IsCorrectDelete(#127, PointsLabeledEdit.Text,

PointsLabeledEdit.SelStart + 1);

If CurKey = #0 Then

Key := 0;

End;

If (Key = VK\_DELETE) And (PointsLabeledEdit.SelText <> '') Then

Begin

CurKey := IsCorrectSelDelete(#127, PointsLabeledEdit.Text,

PointsLabeledEdit.SelText,

PointsLabeledEdit.SelStart);

If CurKey = #0 Then

Key := 0;

End;

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

AddRecordForm.Close;

End;

Procedure TAddRecordForm.PointsLabeledEditKeyPress(Sender: TObject;

Var Key: Char);

Begin

Key := CheckInput(Key, PointsLabeledEdit);

End;

Procedure TAddRecordForm.ReferenceButtonClick(Sender: TObject);

Var

InstractionTest: String;

Begin

InstractionTest := ' - В первых трёх полях ограничений на ввод нет за

исключением' + #13#10 + , что максимальная длина записи 20

символов. Опасайтесь' + #13#10 + 'пробелов, они не видны в

турнирной таблице.' + #13#10 + ' - У поля итогового

рейтинга команды есть ограничения: ' + #13#10 + 'Там можно

писать только натуральные числа в диапазоне от 0' + #13#10

+ 'до 100.' + #13#10 + ' - После того как все поля

заполнены нажимайте на кнопку' + #13#10 + '''' +

AddButton.Caption + ''';';

CreateModalForm('Справка', InstractionTest, Screen.Width \* 27 Div 100,

Screen.Height \* 21 Div 100);

End;

Procedure TAddRecordForm.TeamNameLabeledEditChange(Sender: TObject);

Begin

AddButton.Enabled := IsVisibleAddOnChange(CountryLabeledEdit,

CoachLabeledEdit, PointsLabeledEdit,

TeamNameLabeledEdit);

End;

Procedure TAddRecordForm.TeamNameLabeledEditContextPopup(Sender: TObject;

MousePos: TPoint; Var

Handled: Boolean);

Begin

Handled := True;

End;

Procedure TAddRecordForm.TeamNameLabeledEditKeyDown(Sender: TObject; Var Key:

Word; Shift: TShiftState);

Begin

If (Key = VK\_DOWN) Or (Key = VK\_RETURN) Then

SelectNext(ActiveControl, True, True);

If Key = VK\_UP Then

SelectNext(ActiveControl, False, True);

If (Key = VK\_ESCAPE) Then

AddRecordForm.Close;

End;

Procedure TLabeledEdit.WMPaste(Var Msg: TMessage);

Begin

If Clipboard.HasFormat(CF\_TEXT) Then

Begin

Try

If (AddRecordForm.ActiveControl = AddRecordForm.PointsLabeledEdit) And

Not IsCorrectPointsClipboard(Clipboard.AsText,

AddRecordForm.PointsLabeledEdit) Then

Raise Exception.Create('Некорректное количество

очков!'#13#10'Максимальное кол-во очков

100 :(');

If (AddRecordForm.ActiveControl = AddRecordForm.CoachLabeledEdit) And

Not IsCorrectStrings(Clipboard.AsText,

AddRecordForm.CoachLabeledEdit) Then

Raise Exception.Create('Некорректная фамилия главного

тренера!'#13#10'Давайте уместимся в 20

символов.');

If (AddRecordForm.ActiveControl = AddRecordForm.TeamNameLabeledEdit)

And Not IsCorrectStrings(Clipboard.AsText,

AddRecordForm.TeamNameLabeledEdit) Then

Raise Exception.Create('Некорректное название

команды!'#13#10'Нужно более короткое

название.');

If (AddRecordForm.ActiveControl = AddRecordForm.CountryLabeledEdit)

And Not IsCorrectStrings(Clipboard.AsText,

AddRecordForm.CountryLabeledEdit) Then

Raise Exception.Create('Слишком длинное название

страны!'#13#10'Используйте сокращения

:)');

Except

On E: Exception Do

Begin

MessageBox(0, PWideChar(E.Message), 'Ошибка', MB\_ICONERROR);

Exit;

End;

End;

End;

Inherited;

End;

End.

**Unit BackendUnit;**

Interface

Uses

System.Math,

Clipbrd,

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Variants,

System.Classes,

Vcl.Graphics,

Vcl.Controls,

Vcl.Forms,

Vcl.Dialogs,

Vcl.Menus,

Vcl.StdCtrls,

Vcl.Mask,

Vcl.Grids,

Vcl.ExtCtrls;

Type

TString = Array [1 .. 20] Of WideChar;

TFootStatsRecord = Record

Country: TString;

Team: TString;

Coach: TString;

Points: Integer;

End;

Const

TEAMS\_COUNT = 32;

Type

TMassive = Array Of Integer;

TFootballMassive = Array [0 .. TEAMS\_COUNT] Of TFootStatsRecord;

TFileLoader = File Of TFootStatsRecord;

Function GetRecordFromFile(CurentRow: Integer): TFootStatsRecord;

Procedure ChangeRecordInFile(Country, Team, Coach: TString; Points, CurentRow:

Integer);

Procedure InputRecordInFile(Country, Team, Coach: TString; Points: Integer);

Procedure InputRecordsInTableGrid();

Procedure SortRecords();

Procedure DeleteRow(DelRow: Integer);

Function IndexRecord(I: Integer; CurentStr: TString): Integer;

Function CreateResultGrid(StrIndex: Integer): String;

Procedure CreateCorrectionFile();

Procedure LoadRecordsInFile();

Procedure LoadRecordsFromFile();

Function StrToWideChar(SourceString: String): TString;

Function WideCharToStr(SourceWideChar: TString): String;

Function IfRecordExist(Country, Coach, Team, Points: String): Boolean;

Var

CurentRecordsCount: Integer = 0;

MainFile: TFileLoader;

CorrectionFile: TFileLoader;

Const

MainFilePath: String = 'MainFile.txt';

CorrectionFilePath: String = 'CorrectionFile.txt';

TempFilePath: String = 'TempFile.txt';

Implementation

Uses

MainFormUnit,

FrontendUnit;

Function GetRecordFromFile(CurentRow: Integer): TFootStatsRecord;

Var

TempRecord: TFootStatsRecord;

I: Integer;

Begin

I := 0;

AssignFile(CorrectionFile, CorrectionFilePath);

Try

Reset(CorrectionFile);

While Not EOF(CorrectionFile) And (I <> CurentRow) Do

Begin

Read(CorrectionFile, TempRecord);

Inc(I);

End;

Finally

Close(CorrectionFile);

End;

GetRecordFromFile := TempRecord;

End;

Procedure ChangeRecordInFile(Country, Team, Coach: TString; Points, CurentRow:

Integer);

Var

TempRecord: TFootStatsRecord;

TempFile: TFileLoader;

I: Integer;

Begin

I := 1;

AssignFile(CorrectionFile, CorrectionFilePath);

AssignFile(TempFile, TempFilePath);

Try

Rewrite(TempFile);

Reset(CorrectionFile);

While Not EOF(CorrectionFile) Do

Begin

Read(CorrectionFile, TempRecord);

If (I = CurentRow) Then

Begin

TempRecord.Country := Country;

TempRecord.Team := Team;

TempRecord.Coach := Coach;

TempRecord.Points := Points;

Write(TempFile, TempRecord);

End

Else

Write(TempFile, TempRecord);

Inc(I);

End;

Finally

Close(CorrectionFile);

Close(TempFile);

End;

DeleteFile(CorrectionFilePath);

RenameFile(TempFilePath, CorrectionFilePath);

End;

Procedure InputRecordInFile(Country, Team, Coach: TString; Points: Integer);

Var

TempRecord: TFootStatsRecord;

TempFile: TFileLoader;

Begin

AssignFile(CorrectionFile, CorrectionFilePath);

AssignFile(TempFile, TempFilePath);

Try

Rewrite(TempFile);

Reset(CorrectionFile);

While Not EOF(CorrectionFile) Do

Begin

Read(CorrectionFile, TempRecord);

Write(TempFile, TempRecord);

End;

TempRecord.Country := Country;

TempRecord.Team := Team;

TempRecord.Coach := Coach;

TempRecord.Points := Points;

Write(TempFile, TempRecord);

Finally

Close(CorrectionFile);

Close(TempFile);

End;

DeleteFile(CorrectionFilePath);

RenameFile(TempFilePath, CorrectionFilePath);

End;

Procedure InputRecordsInTableGrid();

Var

TempRecord: TFootStatsRecord;

I: Integer;

Begin

AssignFile(CorrectionFile, CorrectionFilePath);

Try

Reset(CorrectionFile);

MainForm.PointTabelStrGrid.RowCount := CurentRecordsCount + 1;

If CurentRecordsCount > 11 Then

MainForm.PointTabelStrGrid.ColWidths[4] :=

(MainForm.PointTabelStrGrid.Width \* 19) div 100;

I := 0;

While Not EOF(CorrectionFile) Do

Begin

Read(CorrectionFile, TempRecord);

InputInCurentRow(I + 1, WideCharToStr(TempRecord.Country),

WideCharToStr(TempRecord.Team),

WideCharToStr(TempRecord.Coach),

TempRecord.Points);

Inc(I);

End;

Finally

Close(CorrectionFile);

End;

End;

Procedure InputFTInRecords(FootballTable: TFootballMassive);

Var

I: Integer;

Begin

AssignFile(CorrectionFile, CorrectionFilePath);

Rewrite(CorrectionFile);

Try

For I := 0 To CurentRecordsCount - 1 Do

Write(CorrectionFile, FootballTable[I]);

Finally

Close(CorrectionFile);

End;

End;

Procedure InputRecordsInFT(Var FootballTable: TFootballMassive);

Var

I: Integer;

Begin

I := 0;

AssignFile(CorrectionFile, CorrectionFilePath);

Reset(CorrectionFile);

Try

While Not EOF(CorrectionFile) Do

Begin

Read(CorrectionFile, FootballTable[I]);

Inc(I);

End;

Finally

Close(CorrectionFile);

End;

End;

Procedure SortRecords();

Var

FootballTable: TFootballMassive;

TempP, I, J: Integer;

TempT, TempCh, TempCy: TString;

Begin

InputRecordsInFT(FootballTable);

For I := 1 To CurentRecordsCount - 1 Do

Begin

TempP := FootballTable[I].Points;

TempT := FootballTable[I].Team;

TempCh := FootballTable[I].Coach;

TempCy := FootballTable[I].Country;

J := I - 1;

While (J >= 0) And (FootballTable[J].Points < TempP) Do

Begin

FootballTable[J + 1].Points := FootballTable[J].Points;

FootballTable[J + 1].Team := FootballTable[J].Team;

FootballTable[J + 1].Coach := FootballTable[J].Coach;

FootballTable[J + 1].Country := FootballTable[J].Country;

FootballTable[J].Points := TempP;

FootballTable[J].Team := TempT;

FootballTable[J].Coach := TempCh;

FootballTable[J].Country := TempCy;

Dec(J);

End;

End;

InputFTInRecords(FootballTable);

End;

Procedure DeleteRow(DelRow: Integer);

Var

TempRecord: TFootStatsRecord;

TempFile: TFileLoader;

I: Integer;

Begin

I := 1;

AssignFile(CorrectionFile, CorrectionFilePath);

AssignFile(TempFile, TempFilePath);

Try

Rewrite(TempFile);

Reset(CorrectionFile);

While Not EOF(CorrectionFile) Do

Begin

Read(CorrectionFile, TempRecord);

If (I <> DelRow) Then

Write(TempFile, TempRecord);

Inc(I);

End;

Finally

Close(CorrectionFile);

Close(TempFile);

End;

DeleteFile(CorrectionFilePath);

RenameFile(TempFilePath, CorrectionFilePath);

End;

Function IndexRecord(I: Integer; CurentStr: TString): Integer;

Var

FootballTable: TFootballMassive;

J: Integer;

Begin

InputRecordsInFT(FootballTable);

IndexRecord := -1;

Case I Of

0:

Begin

For J := Low(FootballTable) To High(FootballTable) Do

If (FootballTable[J].Country = CurentStr) Then

Begin

IndexRecord := J;

Exit;

End;

End;

1:

Begin

For J := Low(FootballTable) To High(FootballTable) Do

If (FootballTable[J].Team = CurentStr) Then

Begin

IndexRecord := J;

Exit;

End;

End;

2:

Begin

For J := Low(FootballTable) To High(FootballTable) Do

If (FootballTable[J].Coach = CurentStr) Then

Begin

IndexRecord := J;

Exit;

End;

End;

3:

Begin

For J := Low(FootballTable) To High(FootballTable) Do

If (IntToStr(FootballTable[J].Points) =

WideCharToStr(CurentStr)) Then

Begin

IndexRecord := J;

Exit;

End;

End;

End;

End;

Function CreateResultGrid(StrIndex: Integer): String;

Var

TempRecord: TFootStatsRecord;

ResStr: String;

Begin

TempRecord := GetRecordFromFile(StrIndex);

ResStr := 'Страна: ' + WideCharToStr(TempRecord.Country) + ';'#13#10;

ResStr := ResStr + 'Название команды: ' + WideCharToStr(TempRecord.Team) +

';'#13#10;

ResStr := ResStr + 'Главный Тренер: ' + WideCharToStr(TempRecord.Coach) +

';'#13#10;

ResStr := ResStr + 'Итоговый результат:' + IntToStr(TempRecord.Points);

CreateResultGrid := ResStr;

End;

Procedure CreateCorrectionFile();

Begin

AssignFile(CorrectionFile, CorrectionFilePath);

Rewrite(CorrectionFile);

Close(CorrectionFile);

End;

Procedure LoadRecordsInFile();

Begin

If FileExists(MainFilePath) Then

DeleteFile(MainFilePath);

RenameFile(CorrectionFilePath, MainFilePath);

End;

Procedure LoadRecordsFromFile();

Var

TempRecord: TFootStatsRecord;

Begin

CreateCorrectionFile;

CurentRecordsCount := 0;

AssignFile(MainFile, MainFilePath);

AssignFile(CorrectionFile, CorrectionFilePath);

Try

Reset(MainFile);

ReWrite(CorrectionFile);

While Not EOF(MainFile) Do

Begin

Read(MainFile, TempRecord);

Write(CorrectionFile, TempRecord);

Inc(CurentRecordsCount);

End;

CloseFile(MainFile);

CloseFile(CorrectionFile);

Except

Rewrite(MainFile);

Close(MainFile);

End;

SortRecords();

InputRecordsInTableGrid();

End;

Function StrToWideChar(SourceString: String): TString;

Var

DestArray: TString;

NumCharsToCopy, I: Integer;

Begin

NumCharsToCopy := Min(Length(SourceString), SizeOf(DestArray) Div

SizeOf(WideChar) - 1);

For I := 1 To NumCharsToCopy Do

DestArray[I] := WideChar(SourceString[I]);

For I := NumCharsToCopy + 1 To High(DestArray) Do

DestArray[I] := WideChar(#0);

StrToWideChar := DestArray;

End;

Function WideCharToStr(SourceWideChar: TString): String;

Var

ResStr: String;

I: Integer;

Begin

ResStr := '';

For I := Low(SourceWideChar) To High(SourceWideChar) Do

Begin

If SourceWideChar[I] <> #0 Then

ResStr := ResStr + String(SourceWideChar[I]);

End;

WideCharToStr := ResStr;

End;

Function IfRecordExist(Country, Coach, Team, Points: String): Boolean;

Var

TempRecord: TFootStatsRecord;

I: Integer;

Begin

For I := 1 To CurentRecordsCount Do

Begin

TempRecord := GetRecordFromFile(I);

If (TempRecord.Country = StrToWideChar(Country)) And (TempRecord.Coach =

StrToWideChar(Coach)) And

(TempRecord.Team = StrToWideChar(Team)) And

(IntToStr(TempRecord.Points) = Points) Then

Begin

IfRecordExist := True;

Exit;

End;

End;

IfRecordExist := False;

End;

End.

**Unit FrontendUnit;**

Interface

Uses

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Variants,

System.Classes,

Vcl.Graphics,

Vcl.Controls,

Vcl.Forms,

Vcl.Dialogs,

Vcl.Buttons,

Vcl.ExtCtrls,

Vcl.StdCtrls,

Vcl.Imaging.Jpeg,

Vcl.Imaging.Pngimage,

System.ImageList,

Vcl.ImgList,

Vcl.BaseImageCollection,

Vcl.ImageCollection,

Vcl.Menus,

Vcl.Grids;

Procedure CreateModalForm(CaptionText, LabelText: String; ModalWidth, ModalHeight:

Integer);

Function IsCorrectPointsClipboard(ClipbrdText: String; PointsLabeledEdit:

TLabeledEdit): Boolean;

Function IsCorrectStrings(ClipbrdText: String; LabeledEdit: TLabeledEdit):

Boolean;

Function IsVisibleAddOnChange(CountryLEdit, CoachLEdit, PointsLEdit,

TeamNameLEdit: TLabeledEdit): Boolean;

Procedure InputInCurentRow(I: Integer; Country, Team, Coach: String; Points:

Integer);

Function CheckKeyCondition(CurentText: String; Key: Char): Char;

Function IsCorrectDelete(Key: Char; CurentText: String; SelStart: Integer): Char;

Function IsCorrectSelDelete(Key: Char; CurentText, SelText: String; SelStart:

Integer): Char;

Function IsCorrectSelTextInputWithKey(Key: Char; CurentText, SelText: String;

SelStart: Integer): Char;

Function CheckInput(Key: Char; PointsLabeledEdit: TLabeledEdit): Char;

Const

MAX\_STR\_LENGTH: Integer = 20;

MIN\_POINTS: Integer = 0;

MAX\_POINTS: Integer = 100;

Implementation

Uses

MainFormUnit,

BackendUnit;

Procedure CreateModalForm(CaptionText, LabelText: String; ModalWidth, ModalHeight:

Integer);

Var

ModalForm: TForm;

ModalLabel: TLAbel;

Begin

ModalForm := TForm.Create(Nil);

Try

ModalForm.Caption := CaptionText;

ModalForm.Width := ModalWidth;

ModalForm.Height := ModalHeight;

ModalForm.Position := PoScreenCenter;

ModalForm.BorderStyle := BsSingle;

ModalForm.BorderIcons := [BiSystemMenu];

ModalForm.FormStyle := FsStayOnTop;

ModalForm.Icon := MainForm.Icon;

ModalLabel := TLabel.Create(ModalForm);

ModalLabel.Parent := ModalForm;

ModalLabel.Caption := LabelText;

ModalLabel.Left := 10;

ModalLabel.Top := 5;

ModalForm.ShowModal;

Finally

ModalForm.Free;

End;

End;

Function IsCorrectPointsClipboard(ClipbrdText: String; PointsLabeledEdit:

TLabeledEdit): Boolean;

Var

Cursor: Integer;

WorkStr, BufStr: String;

IsCorrect: Boolean;

Begin

IsCorrect := True;

PointsLabeledEdit.ClearSelection;

Cursor := PointsLabeledEdit.SelStart;

WorkStr := PointsLabeledEdit.Text;

Insert(ClipbrdText, WorkStr, Cursor);

Try

IsCorrect := Not((StrToInt(WorkStr) > MAX\_POINTS) Or (StrToInt(WorkStr) <

MIN\_POINTS));

IsCorrect := IsCorrect And (Length(WorkStr) > 1) And (WorkStr[1] = '0');

Except

IsCorrect := False;

End;

IsCorrectPointsClipboard := IsCorrect;

End;

Function IsCorrectStrings(ClipbrdText: String; LabeledEdit: TLabeledEdit):

Boolean;

Var

Cursor: Integer;

WorkStr: String;

IsCorrect: Boolean;

Begin

LabeledEdit.ClearSelection;

Cursor := LabeledEdit.SelStart;

WorkStr := LabeledEdit.Text;

Insert(ClipbrdText, WorkStr, Cursor);

Try

IsCorrect := Not(Length(WorkStr) > MAX\_STR\_LENGTH);

Except

IsCorrect := False;

End;

IsCorrectStrings := IsCorrect;

End;

Function IsVisibleAddOnChange(CountryLEdit, CoachLEdit, PointsLEdit,

TeamNameLEdit: TLabeledEdit): Boolean;

Var

IsCorrect: Boolean;

Begin

Try

IsCorrect := (CountryLEdit.Text <> '') And (CoachLEdit.Text <> '') And

(PointsLEdit.Text <> '') And (TeamNameLEdit.Text <> '');

Except

IsCorrect := False;

End;

IsVisibleAddOnChange := IsCorrect;

End;

Procedure InputInCurentRow(I: Integer; Country, Team, Coach: String; Points:

Integer);

Begin

MainForm.PointTabelStrGrid.Cells[0, I] := IntToStr(I);

MainForm.PointTabelStrGrid.Cells[1, I] := Country;

MainForm.PointTabelStrGrid.Cells[2, I] := Team;

MainForm.PointTabelStrGrid.Cells[3, I] := Coach;

MainForm.PointTabelStrGrid.Cells[4, I] := IntToStr(Points);

End;

Function CheckKeyCondition(CurentText: String; Key: Char): Char;

Begin

Try

If CurentText <> '' Then

If (StrToInt(CurentText) > MAX\_POINTS) Or (StrToInt(CurentText) <

MIN\_POINTS) Then

Key := #0;

If (Length(CurentText) > 0) And (CurentText[1] = '0') Then

Key := #0;

If (Length(CurentText) > 1) And (CurentText[1] = '-') And

(CurentText[2] = '0') Then

Key := #0;

Except

Key := #0;

End;

CheckKeyCondition := Key;

End;

Function IsCorrectDelete(Key: Char; CurentText: String; SelStart: Integer): Char;

Begin

Delete(CurentText, SelStart, 1);

IsCorrectDelete := CheckKeyCondition(CurentText, Key);

End;

Function IsCorrectSelDelete(Key: Char; CurentText, SelText: String; SelStart:

Integer): Char;

Begin

Delete(CurentText, SelStart + 1, Length(SelText));

IsCorrectSelDelete := CheckKeyCondition(CurentText, Key);

End;

Function IsCorrectSelTextInputWithKey(Key: Char; CurentText, SelText: String;

SelStart: Integer): Char;

Begin

Delete(CurentText, SelStart + 1, Length(SelText));

Insert(Key, CurentText, SelStart + 1);

IsCorrectSelTextInputWithKey := CheckKeyCondition(CurentText, Key);

End;

Function CheckInput(Key: Char; PointsLabeledEdit: TLabeledEdit): Char;

Const

GOOD\_VALUES = ['0' .. '9', #08, #$16];

Begin

If Not CharInSet(Key, GOOD\_VALUES) Then

Key := #0;

If (PointsLabeledEdit.SelText <> '') Then

Key := IsCorrectSelTextInputWithKey(Key, PointsLabeledEdit.Text,

PointsLabeledEdit.SelText,

PointsLabeledEdit.SelStart);

If (PointsLabeledEdit.Text = '10') And ((Key <> '0') And (Key <> #08)) Then

Key := #0;

If (Length(PointsLabeledEdit.Text) = PointsLabeledEdit.MaxLength - 1) And

(PointsLabeledEdit.Text <> '10') And (Key <> #08) Then

Key := #0;

If (Length(PointsLabeledEdit.Text) > 0) And (PointsLabeledEdit.Text[1] = '0')

And (PointsLabeledEdit.SelStart = 0) And (Key = '0') Then

Key := #0;

If (PointsLabeledEdit.Text = '0') And (Key <> #08) And

(PointsLabeledEdit.SelStart = 1) Then

Key := #0;

If (Key = #08) And (PointsLabeledEdit.SelText = '') Then

Key := IsCorrectDelete(Key, PointsLabeledEdit.Text,

PointsLabeledEdit.SelStart);

If (Key = #08) And (PointsLabeledEdit.SelText <> '') Then

Key := IsCorrectSelDelete(Key, PointsLabeledEdit.Text,

PointsLabeledEdit.SelText,

PointsLabeledEdit.SelStart);

CheckInput := Key;

End;

End.

Код программы на **C#**:

using System;

using System.Text;

using System.Xml;

using static System.Runtime.InteropServices.JavaScript.JSType;

namespace Proj4\_2 {

enum PanelFunctions {

ADD,

CHANGE,

SEARCH,

DELETE,

REFERENCE,

EXIT

}

enum recordPoints {

COUNTRY,

TEAM,

COACH,

POINTS,

REFERENCE,

EXIT

}

class MainClass() {

record struct TFootballRecord(string Country, string Team, string Coach,

int Points);

static TFootballRecord[] recordArray = new TFootballRecord[0];

const int MAX\_STR\_LENGTH = 20;

const int MAX\_N = 100;

const int MIN\_N = 0;

const int MAX\_RECORDS = 32;

static string MainFilePath = "MainFile.txt";

static string CorrectionFilePath = "CorrectionFile.txt";

static string TempFilePath = "TempFile.txt";

static string[] mainOptions = { " < Добавить запись > ",

" < Изменить запись > ",

" < Найти запись > ",

" < Удалить запись > ",

" < Справка >", " < Выход > " };

static string[] changeRecordOptions = { "Страна", "Команда", "Тренер",

"Рейтинг", "Справка",

"<- Вернуться" };

static string[] exitOption = { "<- Вернуться" };

static void showReference() {

Console.Clear();

string prompt = $"""

Инструкция:

1. Чтобы добавить новую запись выберите кнопку {mainOptions[0]};

2. Если вы хотите изменить какую-то запись выберите

{mainOptions[1]};

3. Если у вас много записей и вы потерялись, выберите

{mainOptions[2]};

4. Если запись необходимо убрать из таблицы, выберите

{mainOptions[3]};

5. Чтобы выйти выбирайте вариант {mainOptions[5]}.

Ограничения по вводу:

1. Поля: {changeRecordOptions[0]}, {changeRecordOptions[1]},

{changeRecordOptions[2]} это

строки максимальной длиной в {MAX\_STR\_LENGTH} символов;

1. Поле {changeRecordOptions[3]} имеет ограничение [{MIN\_N};

{MAX\_N}].

Работа с файлами:

В форме автоматически реализованы типизированный файл и файл

корректур. Если вы сделали запись и закрыли форму, то всё

сохранится автоматически и при запуске будет выгружено.

Максимальное колличество команд, которые можно записать в таблицу:

{MAX\_RECORDS}.

В турнирной таблице записи упорядочены по убыванию рейтинга

команд.

В случае когда набрано одинаковое кол-во очков, первой будет идти

команда, которая была записана в таблицу раньше.

""";

getSelectedIndex(prompt, exitOption);

}

static void importRecordsForSort() {

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

Array.Resize(ref recordArray, recordArray.Length + 1);

recordArray[recordArray.Length - 1].Country = temp.Country;

recordArray[recordArray.Length - 1].Team = temp.Team;

recordArray[recordArray.Length - 1].Coach = temp.Coach;

recordArray[recordArray.Length - 1].Points = temp.Points;

}

}

}

static void importSortRecInFile() {

using (BinaryWriter writer = new

BinaryWriter(File.Open(CorrectionFilePath, FileMode.OpenOrCreate)))

for (int i = 0; i < recordArray.Length; ++i) {

writer.Write(recordArray[i].Country);

writer.Write(recordArray[i].Team);

writer.Write(recordArray[i].Coach);

writer.Write(recordArray[i].Points);

}

}

static void sortRecords() {

int j, points;

TFootballRecord[] temp = new TFootballRecord[1];

recordArray = new TFootballRecord[0];

importRecordsForSort();

for (int I = 1; I < recordArray.Length; I++) {

temp[0] = recordArray[I];

points = recordArray[I].Points;

j = I - 1;

while (j >= 0 && recordArray[j].Points < points) {

recordArray[j + 1] = recordArray[j];

recordArray[j] = temp[0];

j--;

}

}

importSortRecInFile();

recordArray = new TFootballRecord[0];

}

static void displayOptions(string prompt, string[] options,

int selectedIndex = 0) {

Console.WriteLine(prompt);

for (int i = 0; i < options.Length; ++i) {

string curentOption = options[i];

string prefix;

if (i == selectedIndex) {

prefix = "\*";

Console.ForegroundColor = ConsoleColor.Black;

Console.BackgroundColor = ConsoleColor.White;

}

else {

prefix = " ";

Console.ForegroundColor = ConsoleColor.White;

Console.BackgroundColor = ConsoleColor.Black;

}

Console.WriteLine($"{prefix} {curentOption}");

}

Console.ResetColor();

}

static int getSelectedIndex(string prompt, string[] options,

int selectedIndex = 0) {

ConsoleKey keyPressed;

do {

Console.Clear();

displayOptions(prompt, options, selectedIndex);

ConsoleKeyInfo keyInfo = Console.ReadKey(true);

keyPressed = keyInfo.Key;

if (keyPressed == ConsoleKey.UpArrow)

if (--selectedIndex == -1) selectedIndex = options.Length - 1;

if (keyPressed == ConsoleKey.DownArrow)

if (++selectedIndex == options.Length) selectedIndex = 0;

} while (keyPressed != ConsoleKey.Enter);

return selectedIndex;

}

static PanelFunctions searchQurentMainMethod(int curentIndex) {

switch (curentIndex) {

case (int)PanelFunctions.ADD: return PanelFunctions.ADD;

case (int)PanelFunctions.CHANGE: return PanelFunctions.CHANGE;

case (int)PanelFunctions.SEARCH: return PanelFunctions.SEARCH;

case (int)PanelFunctions.DELETE: return PanelFunctions.DELETE;

case (int)PanelFunctions.REFERENCE: return

PanelFunctions.REFERENCE;

default: return PanelFunctions.EXIT;

}

}

static string outputTable() {

int columnWidth = 20;

string formatString = " {0,-" + columnWidth + "} {1,-" +

columnWidth + "} {2,-" + columnWidth + "}

{3,-" + columnWidth + "} {4,-" +

columnWidth + "}";

string prompt = string.Format(formatString, "Место", "Страна",

"Команда", "Тренер", "Рейтинг");

prompt += '\n' + new string('-', (columnWidth + 1) \* 5);

int i = 1;

using (BinaryReader reader = new

BinaryReader(File.Open (CorrectionFilePath, FileMode.OpenOrCreate))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

prompt += '\n' + string.Format(formatString, i++,

temp.Country, temp.Team,

temp.Coach, temp.Points);

}

}

return prompt;

}

static string inputString(string prompt, in int maxLength) {

Console.Write(prompt);

bool isCorrect = false;

string curentStr = string.Empty;

do {

curentStr = Console.ReadLine() ?? string.Empty;

isCorrect = !((curentStr.Length > maxLength) ||

(curentStr == string.Empty));

if (!isCorrect) Console.Error.Write($"Ошибка! Попробуйте снова

(максимальная длина

{maxLength} символов): ");

} while (!isCorrect);

return curentStr;

}

static int inputNum(string prompt, in int max, in int min) {

Console.Write(prompt);

bool isCorrect = true;

int num = 0;

do {

try {

num = Convert.ToInt32(Console.ReadLine());

isCorrect = true;

}

catch {

isCorrect = false;

}

isCorrect = isCorrect && !(num > max) && !(num < min);

if (!isCorrect) Console.Error.Write($"Ошибка! Попробуйте снова

[{min}; {max}]: ");

} while (!isCorrect);

return num;

}

static void addRecordInFile(TFootballRecord newRecord) {

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate)))

using (BinaryWriter writer = new

BinaryWriter(File.Open(TempFilePath, FileMode.CreateNew))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length)

{

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

writer.Write(temp.Country);

writer.Write(temp.Team);

writer.Write(temp.Coach);

writer.Write(temp.Points);

}

writer.Write(newRecord.Country);

writer.Write(newRecord.Team);

writer.Write(newRecord.Coach);

writer.Write(newRecord.Points);

}

File.Delete(CorrectionFilePath);

File.Move(TempFilePath, CorrectionFilePath);

}

static void addNewRecord() {

string prompt = string.Empty;

Console.Clear();

if (recordArray.Length == MAX\_RECORDS) prompt = "Вы достигли

максимального

колличесвта команд.";

else {

TFootballRecord newRecord = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

Console.WriteLine("Давайте создадим новую запись.");

string country = inputString("Страна: ", MAX\_STR\_LENGTH);

newRecord.Country = country;

string team = inputString("Название команда: ", MAX\_STR\_LENGTH);

newRecord.Team = team;

string coach = inputString("Фамилия гл. тренера: ",

MAX\_STR\_LENGTH);

newRecord.Coach = coach;

int points = inputNum("Рейтинг команды (кол-во очков): ", MAX\_N,

MIN\_N);

newRecord.Points = points;

addRecordInFile(newRecord);

sortRecords();

prompt = $"""

Страна: {country};

Команда: {team};

Тренер: {coach};

Рейтинг: {points};

""";

}

getSelectedIndex(prompt, exitOption);

}

static string[] createRecordsOptions(int columnWidth, string formatString)

{

string[] result = new string[0];

int i = 1;

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

Array.Resize(ref result, result.Length + 1);

result[i - 1] += string.Format(formatString, i++,

temp.Country, temp.Team,

temp.Coach, temp.Points);

}

}

Array.Resize(ref result, result.Length + 1);

result[i - 1] = exitOption[0];

return result;

}

static int getSelectedRow() {

int columnWidth = 20;

string formatString = "{0,-" + columnWidth + "} {1,-" +

columnWidth + "} {2,-" + columnWidth + "}

{3,-" + columnWidth + "} {4,-" +

columnWidth + "}";

string[] recordsOption = createRecordsOptions(columnWidth,

formatString);

formatString = " {0,-" + columnWidth + "} {1,-" + columnWidth + "}

{2,-" + columnWidth + "} {3,-" + columnWidth + "}

{4,-" + columnWidth + "}";

string prompt = string.Format(formatString, "Место", "Страна",

"Команда", "Тренер", "Рейтинг");

prompt += "\n" + new string('-', (columnWidth + 1) \* 5);

return getSelectedIndex(prompt, recordsOption);

}

static recordPoints qurentPoint(int selectedIndex) {

switch (selectedIndex) {

case (int)recordPoints.COUNTRY: return recordPoints.COUNTRY;

case (int)recordPoints.TEAM: return recordPoints.TEAM;

case (int)recordPoints.COACH: return recordPoints.COACH;

case (int)recordPoints.POINTS: return recordPoints.POINTS;

case (int)recordPoints.REFERENCE: return recordPoints.REFERENCE;

default: return recordPoints.EXIT;

}

}

static void workWithPoint(recordPoints selectedRecordPoint,

ref TFootballRecord curentRow) {

switch (selectedRecordPoint) {

case recordPoints.COUNTRY:

string countrty = inputString("Страна: ", MAX\_STR\_LENGTH);

curentRow.Country = countrty;

break;

case recordPoints.TEAM:

string team = inputString("Название команды: ",

MAX\_STR\_LENGTH);

curentRow.Team = team;

break;

case recordPoints.COACH:

string coach = inputString("Фамилия гл. тренера: ",

MAX\_STR\_LENGTH);

curentRow.Coach = coach;

break;

case recordPoints.POINTS:

int points = inputNum("Рейтинг команды (кол-во очков): ",

MAX\_N, MIN\_N);

curentRow.Points = points;

break;

case recordPoints.REFERENCE:

showReference();

break;

}

}

static void searchCurentRowInFile(int selectedRow,

ref TFootballRecord curentRow,

ref int recordCounter) {

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

if (recordCounter == selectedRow) {

curentRow.Country = temp.Country;

curentRow.Team = temp.Team;

curentRow.Coach = temp.Coach;

curentRow.Points = temp.Points;

}

recordCounter++;

}

}

}

static void inputChangeRowInFile(int selectedRow,

TFootballRecord curentRow) {

int recordCounter = 0;

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate)))

using (BinaryWriter writer = new BinaryWriter(File.Open(TempFilePath,

FileMode.CreateNew))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

if (recordCounter == selectedRow) {

writer.Write(curentRow.Country);

writer.Write(curentRow.Team);

writer.Write(curentRow.Coach);

writer.Write(curentRow.Points);

}

else {

writer.Write(temp.Country);

writer.Write(temp.Team);

writer.Write(temp.Coach);

writer.Write(temp.Points);

}

recordCounter++;

}

}

File.Delete(CorrectionFilePath);

File.Move(TempFilePath, CorrectionFilePath);

}

static void workWithSelectedRow(int selectedRow) {

int recordsCounter = 0;

TFootballRecord curentRow = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

searchCurentRowInFile(selectedRow, ref curentRow, ref recordsCounter);

if (selectedRow == recordsCounter) return;

recordPoints selectedRecordPoint = recordPoints.EXIT;

do {

int columnWidth = 20;

string formatString = "{0,-" + columnWidth + "}

{1,-" + columnWidth + "} {2,-" +

columnWidth + "} {3,-" + columnWidth + "}

{4,-" + columnWidth + "}";

string prompt = string.Format(formatString, "Место", "Страна",

"Команда", "Тренер", "Рейтинг");

prompt += "\n" + new string('-', columnWidth \* 5);

prompt += '\n' + string.Format(formatString, -1,

curentRow.Country, curentRow.Team,

curentRow.Coach, curentRow.Points);

prompt += "\nЧто вы хотите изменить:";

selectedRecordPoint = qurentPoint(getSelectedIndex(prompt,

changeRecordOptions));

workWithPoint(selectedRecordPoint,ref curentRow);

} while (selectedRecordPoint != recordPoints.EXIT);

inputChangeRowInFile(selectedRow, curentRow);

sortRecords();

}

static void changeRecord() {

int selectedRow = 0;

do {

Console.Clear();

selectedRow = getSelectedRow();

workWithSelectedRow(selectedRow);

} while (selectedRow != recordArray.Length);

}

static int returnRow(recordPoints selectedRecordPoint) {

importRecordsForSort();

switch (selectedRecordPoint) {

case recordPoints.POINTS:

int points = inputNum("Введите рейтинг команды

(кол-во очков): ", MAX\_N, MIN\_N);

for (int i = 0; i < recordArray.Length; ++i)

if (recordArray[i].Points == points)

{ return i; }

break;

case recordPoints.COUNTRY:

string country = inputString("Введите страну: ",

MAX\_STR\_LENGTH);

for (int i = 0; i < recordArray.Length; ++i)

if (recordArray[i].Country == country)

{ return i; }

break;

case recordPoints.TEAM:

string team = inputString("Введите название команды: ",

MAX\_STR\_LENGTH);

for (int i = 0; i < recordArray.Length; ++i)

if (recordArray[i].Team == team)

{ return i; }

break;

case recordPoints.COACH:

string coach = inputString("Введите фамилию гл. тренера: ",

MAX\_STR\_LENGTH);

for (int i = 0; i < recordArray.Length; ++i)

if (recordArray[i].Coach == coach)

{ return i; }

break;

}

return -1;

}

static void workWithSearchPoint(recordPoints selectedRecordPoint) {

string prompt = string.Empty;

recordArray = new TFootballRecord[0];

importRecordsForSort();

int recordIndex = returnRow(selectedRecordPoint);

switch (selectedRecordPoint) {

case recordPoints.EXIT: break;

case recordPoints.REFERENCE: showReference(); break;

default:

if (recordIndex == -1) prompt = "Не нашлось записис по

заданному параметру...";

else {

int columnWidth = 20;

string formatString = "{0,-" + columnWidth + "}

{1,-" + columnWidth + "}

{2,-" + columnWidth + "} {3,-" +

columnWidth + "} {4,-" +

columnWidth + "}";

prompt = string.Format(formatString, "Место", "Страна",

"Команда", "Тренер", "Рейтинг");

prompt += "\n" + new string('-', columnWidth \* 5);

prompt += '\n' + string.Format(formatString, 0,

recordArray[recordIndex].Country,

recordArray[recordIndex].Team,

recordArray[recordIndex].Coach,

recordArray[recordIndex].Points);

}

getSelectedIndex(prompt, exitOption);

break;

}

recordArray = new TFootballRecord[0];

}

static void searchRecord() {

Console.Clear();

recordPoints selectedRecordPoint = recordPoints.EXIT;

do {

string prompt = "Какой параметр выберем: ";

selectedRecordPoint = qurentPoint(getSelectedIndex(prompt,

changeRecordOptions));

workWithSearchPoint(selectedRecordPoint);

} while (selectedRecordPoint != recordPoints.EXIT);

}

static void tryDeleteRow(int deleteRow, ref int upperLimit) {

int i = 0;

using (BinaryReader reader = new

BinaryReader(File.Open(CorrectionFilePath, FileMode.OpenOrCreate)))

using (BinaryWriter writer = new

BinaryWriter(File.Open(TempFilePath, FileMode.CreateNew))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length)

{

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

if (i++ != deleteRow) {

writer.Write(temp.Country);

writer.Write(temp.Team);

writer.Write(temp.Coach);

writer.Write(temp.Points);

}

}

}

File.Delete(CorrectionFilePath);

File.Move(TempFilePath, CorrectionFilePath);

upperLimit = i;

}

static void deleteRecord() {

int deleteRow = 0;

int upperLimit = 0;

do {

Console.Clear();

deleteRow = getSelectedRow();

tryDeleteRow(deleteRow, ref upperLimit);

} while (deleteRow != upperLimit);

}

static void workWithMethod(PanelFunctions curentMethod) {

switch (curentMethod) {

case PanelFunctions.ADD: addNewRecord(); break;

case PanelFunctions.CHANGE: changeRecord(); break;

case PanelFunctions.SEARCH: searchRecord(); break;

case PanelFunctions.DELETE: deleteRecord(); break;

case PanelFunctions.REFERENCE: showReference(); break;

}

}

static void importRecordsFromFile() {

if (!File.Exists(MainFilePath)) using

(FileStream fs = File.Create(MainFilePath));

using (BinaryReader reader = new BinaryReader(File.Open(MainFilePath,

FileMode.OpenOrCreate)))

using (BinaryWriter writer = new

BinaryWriter(File.Open(CorrectionFilePath, FileMode.CreateNew))) {

TFootballRecord temp = new TFootballRecord(string.Empty,

string.Empty,

string.Empty, 0);

while (reader.BaseStream.Position < reader.BaseStream.Length) {

temp.Country = reader.ReadString();

temp.Team = reader.ReadString();

temp.Coach = reader.ReadString();

temp.Points = reader.ReadInt32();

writer.Write(temp.Country);

writer.Write(temp.Team);

writer.Write(temp.Coach);

writer.Write(temp.Points);

}

}

}

static void exportRecordsInFile() {

File.Delete(MainFilePath);

File.Move(CorrectionFilePath, MainFilePath);

}

static void programBlock() {

importRecordsFromFile();

PanelFunctions curentMethod = 0;

do {

string prompt = outputTable();

prompt += '\n';

curentMethod = searchQurentMainMethod(getSelectedIndex(prompt,

mainOptions));

workWithMethod(curentMethod);

} while (curentMethod != PanelFunctions.EXIT);

exportRecordsInFile();

Environment.Exit(0);

}

public static void Main(string[] args) {

Encoding.RegisterProvider(CodePagesEncodingProvider.Instance);

Console.OutputEncoding = Encoding.GetEncoding(1251);

Console.InputEncoding = Encoding.GetEncoding(1251);

Console.Title = "Футбольный справочник™";

programBlock();

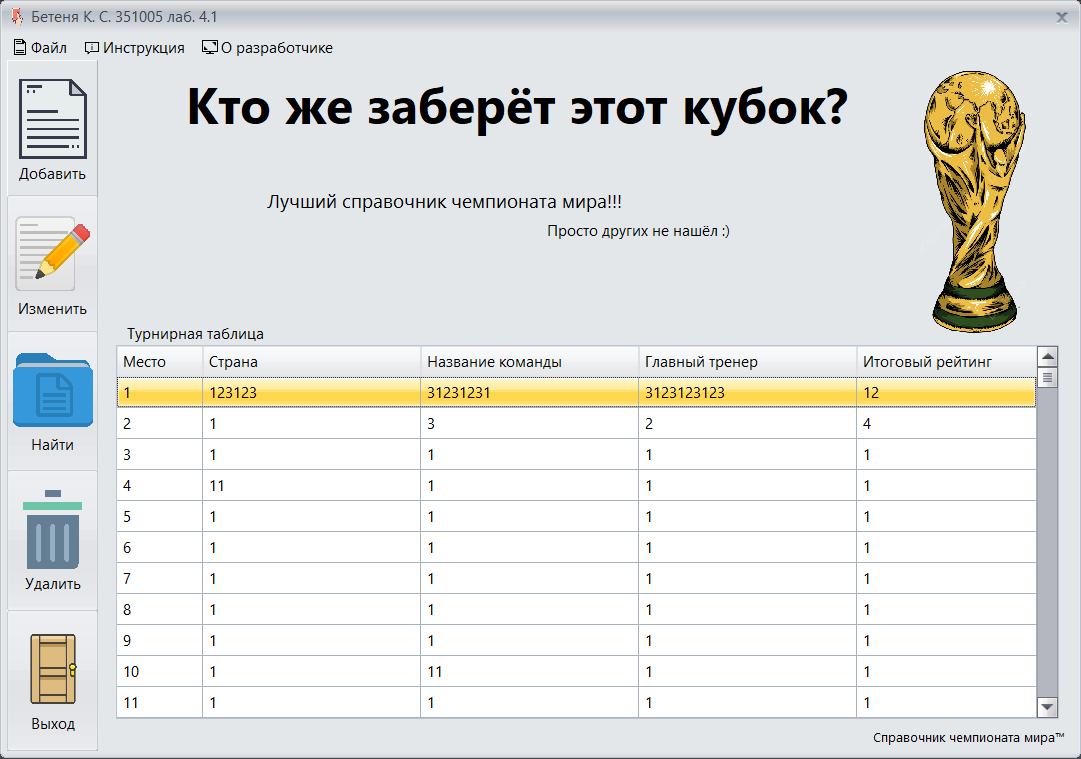
Console.ReadKey(true);

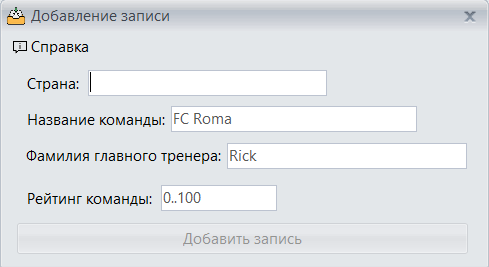
}

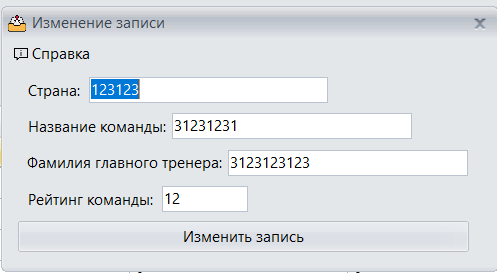
}

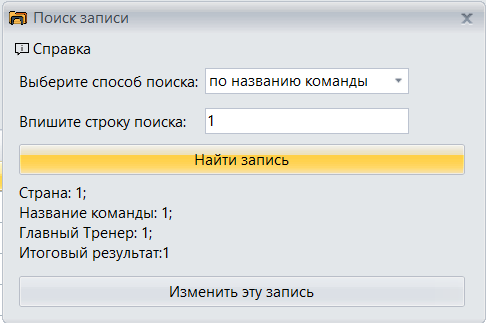
}

Результат в **Delph:**

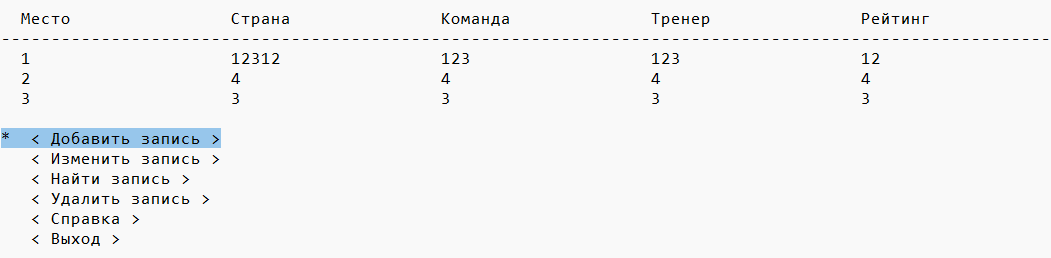


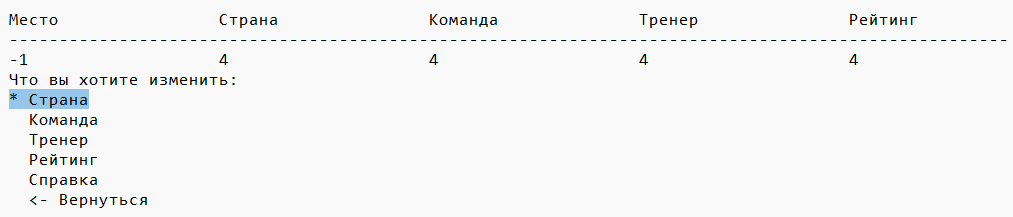






Результат в **C#**:





Блок-схема:

