**УО «Белорусский государственный университет информатики и радиоэлектроники»**

**Кафедра ПОИТ**

**Отчёт по лабораторной работе №5.2**

**По предмету**

**Основы алгоритмизации и программирования**

**Вариант 6**

**Выполнил:**

**Гладкий М.Г.**

**Проверила:**

**Данилова Г.В.**

**Группа 851001**

**Минск 2019**

**Задание:**

Комбинаторика. Обойти шахматную доску ходом коня, так, чтобы все клетки были пройдены по одному разу. Визуализация!

**Delphi 10:**

**unit** MainUn;

**interface**

**uses**

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

System.Classes, Vcl.Graphics, Vcl.Controls, Vcl.Forms, Vcl.Dialogs,

Vcl.Menus, Vcl.ExtCtrls, Vcl.StdCtrls, Vcl.Grids;

**type**

TBoard = **array** [0..7] **of array** [0..7] **of** Integer;

TMain = **class**(TForm)

SG: TStringGrid;

Menu: TMainMenu;

BlankPopUp: TPopupMenu;

btnPush: TButton;

btnRetry: TButton;

btnAuto: TButton;

Tick: TTimer;

Info: TMenuItem;

Developer: TMenuItem;

**Exit**: TMenuItem;

**function** FindPossibleRoutes (X, Y: Integer; Board: TBoard): Integer;

**procedure** InfoClick(Sender: TObject);

**procedure** DeveloperClick(Sender: TObject);

**procedure** ExitClick(Sender: TObject);

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

**procedure** FormCreate(Sender: TObject);

**procedure** btnAutoClick(Sender: TObject);

**procedure** FormKeyPress(Sender: TObject; **var** Key: Char);

**procedure** TickTimer(Sender: TObject);

**procedure** btnPushClick(Sender: TObject);

**procedure** btnRetryClick(Sender: TObject);

**procedure** SGDrawCell(Sender: TObject; ACol, ARow: Integer; Rect:

TRect; State: TGridDrawState);

**procedure** SGSelectCell(Sender: TObject; ACol, ARow: Integer;

**var** CanSelect: Boolean);

**private**

{ Private declarations }

**public**

{ Public declarations }

**end**;

**var**

Main: TMain;

**implementation**

{$R \*.dfm}

**var**

X, Y, EndCounter: Integer;

Board: TBoard;

**procedure** TMain.btnAutoClick(Sender: TObject);

**begin**

Tick.Enabled := True;

btnAuto.Enabled := False;

**end**;

**procedure** TMain.btnPushClick(Sender: TObject);

**var**

i, j: Integer;

**begin**

Inc(EndCounter);

**if** (EndCounter < 65) **then**

**begin**

**for** i := 0 **to** 7 **do**

**for** j := 0 **to** 7 **do**

**begin**

**if** (Board[i][j] = EndCounter) **then**

SG.Cells[j, i] := 'К';

**if** (Board[i][j] = EndCounter - 1) **then**

SG.Cells[j, i] := '\*';

**end**;

**end**;

**if** (EndCounter = 64) **then**

**begin**

btnPush.Enabled := False;

Tick.Enabled := False;

btnAuto.Enabled := False;

MessageDlg('Bypass is over.', mtInformation, [mbOk], 0);

**end**;

**end**;

**procedure** TMain.btnRetryClick(Sender: TObject);

**var**

i, j: Integer;

Col, Row: Integer;

Rect: TRect;

**begin**

**with** SG.Canvas **do**

**for** i := 0 **to** 7 **do**

**for** j := 0 **to** 7 **do**

**begin**

Brush.Color := clWhite;

Rect := SG.CellRect(j, i);

FillRect(Rect);

**end**;

EndCounter := 1;

btnPush.Enabled := True;

btnAuto.Enabled := False;

Tick.Enabled := False;

**with** SG **as** TStringGrid **do**

**begin**

**for** i := 0 **to** 7 **do**

**for** j := 0 **to** 7 **do**

Cells[j, i] := '';

Enabled := True;

**end**;

**end**;

**function** TMain.FindPossibleRoutes (X, Y: Integer; Board: TBoard): Integer;

**var**

Counter: Integer;

**begin**

Counter := 0;

**if** (X > 0) **and** (Y > 1) **and** (Board[X - 1][Y - 2] = 0) **then**

Inc(Counter);

**if** (X > 0) **and** (Y < 6) **and** (Board[X - 1][Y + 2] = 0) **then**

Inc(Counter);

**if** (X > 1) **and** (Y > 0) **and** (Board[X - 2][Y - 1] = 0) **then**

Inc(Counter);

**if** (X > 1) **and** (Y < 7) **and** (Board[X - 2][Y + 1] = 0) **then**

Inc(Counter);

**if** (X < 7) **and** (Y > 1) **and** (Board[X + 1][Y - 2] = 0) **then**

Inc(Counter);

**if** (X < 7) **and** (Y < 6) **and** (Board[X + 1][Y + 2] = 0) **then**

Inc(Counter);

**if** (X < 6) **and** (Y > 0) **and** (Board[X + 2][Y - 1] = 0) **then**

Inc(Counter);

**if** (X < 6) **and** (Y < 7) **and** (Board[X + 2][Y + 1] = 0) **then**

Inc(Counter);

FindPossibleRoutes := Counter;

**end**;

**procedure** TMain.DeveloperClick(Sender: TObject);

**begin**

MessageDlg('Developer: Maksim Hladki, 851001', mtInformation, [mbOk], 0);

**end**;

**procedure** TMain.ExitClick(Sender: TObject);

**begin**

close;

**end**;

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

**var**

ButtonSelected: Byte;

**begin**

ButtonSelected := MessageDlg('Are you sure you want to exit?',

mtConfirmation, [mbYes,mbNo], 0);

**if** ButtonSelected <> mrYes **then**

CanClose := False;

**end**;

**procedure** TMain.FormCreate(Sender: TObject);

**begin**

MessageDlg('Choose starting position.', mtInformation, [mbOk], 0);

EndCounter := 1;

**end**;

**procedure** TMain.FormKeyPress(Sender: TObject; **var** Key: Char);

**begin**

**if** (Key = #13) **and** (btnPush.Enabled) **then**

btnPushClick(Sender)

**else**

Key := #0;

**end**;

**procedure** TMain.InfoClick(Sender: TObject);

**begin**

MessageDlg('This program is knight chess bypass. You can choose position

where to start.', mtInformation, [mbOk], 0);

**end**;

**procedure** TMain.SGDrawCell(Sender: TObject; ACol, ARow: Integer; Rect: TRect;

State: TGridDrawState);

**begin**

**with** SG.Canvas **do**

**begin**

**if** (SG.Cells[ACol, ARow] = '\*') **then**

**begin**

**if** EndCounter = 2 **then**

**begin**

Brush.Color := clGreen;

FillRect(Rect);

**end**

**else**

**begin**

Brush.Color := clPurple;

FillRect(Rect);

**end**;

Font := Main.Font;

Font.Size := 13;

Font.Color := clWhite;

TextOut(Rect.Left, Rect.Top, IntToStr(EndCounter - 1));

**end**;

**if** (SG.Cells[ACol, ARow] = 'К') **then**

**begin**

Brush.Color := clSilver;

FillRect(Rect);

Font := Main.Font;

Font.Size := 13;

Font.Style := Font.Style + [fsBold];

TextOut(Rect.Left, Rect.Top, 'К');

**end**;

**end**;

**end**;

**procedure** TMain.SGSelectCell(Sender: TObject; ACol, ARow: Integer;

**var** CanSelect: Boolean);

**var**

i, j, Step, Buff, Buff2: Integer;

**begin**

btnAuto.Enabled := True;

SG.Enabled := False;

X := ARow;

Y := ACol;

SG.Cells[ACol, ARow] := 'К';

Step := 0;

**for** i := 0 **to** 7 **do**

**for** j := 0 **to** 7 **do**

Board[i][j] := 0;

**while** (Step < 64) **do**

**begin**

Inc(Step);

Board[X][Y] := Step;

Buff := 9;

**if** ((X > 0) **and** (Y > 1) **and** (Board[X - 1][Y - 2] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X - 1, Y - 2, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X - 1;

j := Y - 2;

Buff := Buff2;

**end**;

**end**;

**if** ((X > 0) **and** (Y < 6) **and** (Board[X - 1][Y + 2] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X - 1, Y + 2, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X - 1;

j := Y + 2;

Buff := Buff2;

**end**;

**end**;

**if** ((X > 1) **and** (Y > 0) **and** (Board[X - 2][Y - 1] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X - 2, Y - 1, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X - 2;

j := Y - 1;

Buff := Buff2;

**end**;

**end**;

**if** ((X > 1) **and** (Y < 7) **and** (Board[X - 2][Y + 1] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X - 2, Y + 1, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X - 2;

j := Y + 1;

Buff := Buff2;

**end**;

**end**;

**if** ((X < 7) **and** (Y > 1) **and** (Board[X + 1][Y - 2] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X + 1, Y - 2, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X + 1;

j := Y - 2;

Buff := Buff2;

**end**;

**end**;

**if** ((X < 7) **and** (Y < 6) **and** (Board[X + 1][Y + 2] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X + 1, Y + 2, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X + 1;

j := Y + 2;

Buff := Buff2;

**end**;

**end**;

**if** ((X < 6) **and** (Y > 0) **and** (Board[X + 2][Y - 1] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X + 2, Y - 1, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X + 2;

j := Y - 1;

Buff := Buff2;

**end**;

**end**;

**if** ((X < 6) **and** (Y < 7) **and** (Board[X + 2][Y + 1] = 0)) **then**

**begin**

Buff2 := FindPossibleRoutes(X + 2, Y + 1, Board);

**if** (Buff > Buff2) **then**

**begin**

i := X + 2;

j := Y + 1;

**end**;

**end**;

X := i;

Y := j;

**end**;

**end**;

**procedure** TMain.TickTimer(Sender: TObject);

**var**

i, j: Integer;

**begin**

Inc(EndCounter);

**for** i := 0 **to** 7 **do**

**for** j := 0 **to** 7 **do**

**begin**

**if** (Board[i][j] = EndCounter) **then**

SG.Cells[j, i] := 'К';

**if** (Board[i][j] = EndCounter - 1) **then**

SG.Cells[j, i] := '\*';

**end**;

**if** (EndCounter = 64) **then**

**begin**

Tick.Enabled := False;

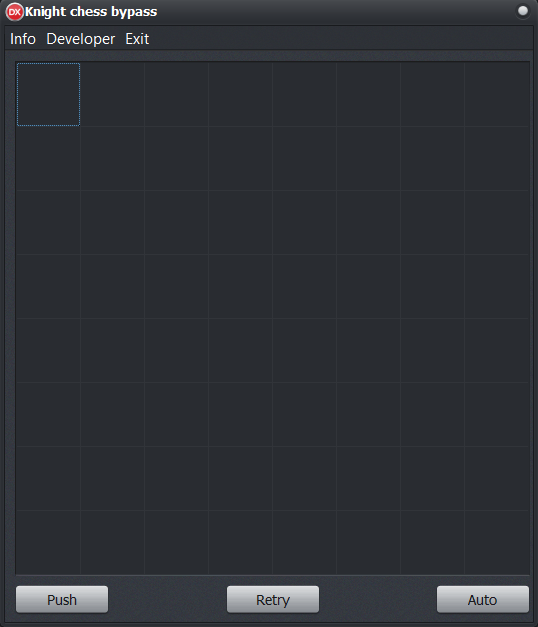
MessageDlg('Bypass is over.', mtInformation, [mbOk], 0);

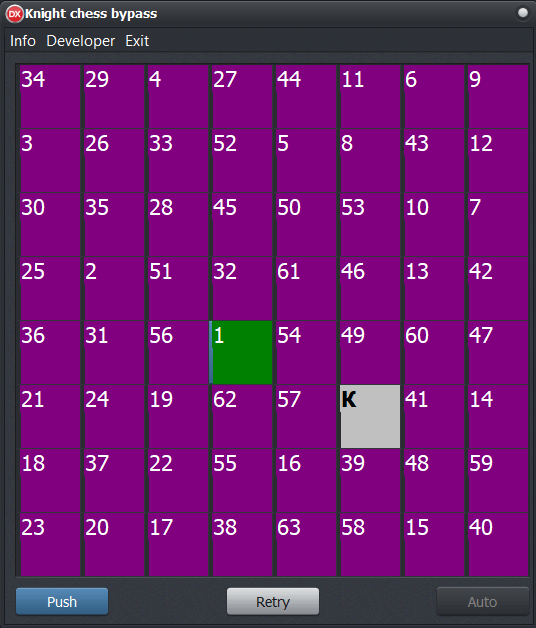
**end**;

**end**;

**end**.

**Скриншоты:**

****

****

**Блок-схема:**





