УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №7.2

по предмету «Основы алгоритмизации и программирования»

Вариант 18

Выполнил:

Машевский Д.В.

Гр. 351003

Проверил:

Данилова Г. В.

Минск 2024

**Задание:**

Преобразовать списки инцидентности в матрицу смежности.

**Код программы Delphi:**

Unit FormsLaba72Unit;

Interface

Uses

Winapi.Windows,

Winapi.Messages,

System.SysUtils,

System.Variants,

System.Classes,

Vcl.Graphics,

Vcl.Controls,

Vcl.Forms,

Vcl.Dialogs,

Vcl.Grids,

Vcl.StdCtrls,

Vcl.Buttons,

Vcl.Menus,

Vcl.ExtCtrls,

Vcl.Imaging.Jpeg;

Const

MAX\_DIGIT = 6;

Type

TLaba72Form = Class(TForm)

MatrixSmezhStringGrid: TStringGrid;

InfoLabel: TLabel;

MainMenu: TMainMenu;

NFile: TMenuItem;

NOpenFile: TMenuItem;

NSaveFile: TMenuItem;

N4: TMenuItem;

EnterSizeBitBtn: TBitBtn;

EnterEdit: TEdit;

InfoGrafLabel: TLabel;

RoflImage: TImage;

PopupMenu: TPopupMenu;

Procedure EnterSizeBitBtnClick(Sender: TObject);

Procedure NOpenFileClick(Sender: TObject);

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

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

Procedure NSaveFileClick(Sender: TObject);

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

Procedure FormCreate(Sender: TObject);

Procedure N4Click(Sender: TObject);

Private

{ Private declarations }

Public

{ Public declarations }

End;

Type

PNode = ^TNode;

TNode = Record

Data: Integer;

Next: PNode;

End;

TArr = Array Of Array Of Integer;

Var

Laba72Form: TLaba72Form;

First: PNode;

AdjacencyMatrix: TArr;

Implementation

{$R \*.dfm}

Function CheckZeroStr(Str: String): Boolean;

Var

I, Len: Integer;

Fl: Boolean;

Begin

Fl := False;

Len := Length(Str) - 1;

For I := 1 To Len Do

Begin

If (Str[I] = '0') And (I <> High(Str)) Then

Begin

If (Str[I + 1] <> '') Then

Fl := True;

End;

End;

CheckZeroStr := Fl;

End;

Function IsNumberInList(Num: Integer): Boolean;

Var

CurrentNode: PNode;

Begin

CurrentNode := First;

While CurrentNode <> Nil Do

Begin

If CurrentNode^.Data = Num Then

Begin

Result := True;

Exit;

End;

CurrentNode := CurrentNode^.Next;

End;

Result := False;

End;

Procedure AddVertexToList(Num: Integer);

Var

NewNode, LastNode: PNode;

Begin

If Not IsNumberInList(Num) Then

Begin

New(NewNode);

NewNode^.Data := Num;

NewNode^.Next := Nil;

If First = Nil Then

First := NewNode

Else

Begin

LastNode := First;

While LastNode^.Next <> Nil Do

LastNode := LastNode^.Next;

LastNode^.Next := NewNode;

End;

End;

End;

Function GetIndex(Num: Integer): Integer;

Var

Index: Integer;

CurrentNode: PNode;

Begin

Index := 0;

CurrentNode := First;

While CurrentNode <> Nil Do

Begin

If CurrentNode^.Data = Num Then

Begin

Result := Index;

Exit;

End;

Inc(Index);

CurrentNode := CurrentNode^.Next;

End;

Result := -1;

End;

Procedure TLaba72Form.CheckEnter(Sender: TObject; Var Key: Char);

Var

Chars: Set Of Char;

EditText: String;

Begin

EditText := (Sender As TEdit).Text;

If (Length(EditText) = 0) Then

Begin

Chars := ['0' .. '9', #8, #31];

End

Else

Chars := ['0' .. '9', #8, #31];

If Not(Key In Chars) Then

Key := #0;

End;

Procedure TLaba72Form.EnterEditKeyPress(Sender: TObject; Var Key: Char);

Var

Value: Integer;

Begin

CheckEnter(Sender, Key);

If Key = #8 Then

If (CheckZeroStr(EnterEdit.Text)) Then

EnterEdit.Text := '';

If TryStrToInt(EnterEdit.Text + Key, Value) Then

If (Value > MAX\_DIGIT) Then

Key := #0;

If EnterEdit.Text = '0' Then

Begin

EnterEdit.Text := '';

End;

End;

Procedure TLaba72Form.EnterSizeBitBtnClick(Sender: TObject);

Var

Len, Temp, I, IndexA, IndexB: Integer;

A, B: TArray<Integer>;

Begin

First := Nil;

Len := StrToInt(EnterEdit.Text);

SetLength(A, Len);

SetLength(B, Len);

For I := 0 To Len - 1 Do

Begin

//Assuming input format is "vertexA vertexB"

//You can replace this with your own input logic based on your form design

A[I] := StrToInt(InputBox('Введите вершину A', 'Введите значение вершины A',

''));

AddVertexToList(A[I]);

B[I] := StrToInt(InputBox('Введите вершину B', 'Введите значение вершины B',

''));

AddVertexToList(B[I]);

End;

SetLength(AdjacencyMatrix, Len, Len);

For I := 0 To Len - 1 Do

Begin

IndexA := GetIndex(A[I]);

IndexB := GetIndex(B[I]);

AdjacencyMatrix[IndexA][IndexB] := 1;

AdjacencyMatrix[IndexB][IndexA] := 1;

End;

//Populate the StringGrid with the adjacency matrix

MatrixSmezhStringGrid.RowCount := Len + 1;

MatrixSmezhStringGrid.ColCount := Len + 1;

For I := 0 To Len - 1 Do

MatrixSmezhStringGrid.Cells[I + 1, 0] := IntToStr(I + 1);

For I := 0 To Len - 1 Do

MatrixSmezhStringGrid.Cells[0, I + 1] := IntToStr(I + 1);

For I := 0 To Len - 1 Do

For Temp := 0 To Len - 1 Do

MatrixSmezhStringGrid.Cells[I + 1, Temp + 1] := IntToStr(AdjacencyMatrix[I][Temp]);

MainMenu.Items[0].Items[1].Enabled := True;

End;

Procedure TLaba72Form.FormCloseQuery(Sender: TObject; Var CanClose: Boolean);

Var

UserChoice: Integer;

Begin

CanClose := False;

UserChoice := MessageBox(Handle, 'Вы уверены, что хотите выйти?', 'Подтверждение', MB\_ICONQUESTION Or MB\_YESNO);

If UserChoice = IDYES Then

CanClose := True;

End;

Procedure TLaba72Form.FormCreate(Sender: TObject);

Begin

MainMenu.Items[0].Items[1].Enabled := False;

End;

Procedure TLaba72Form.N4Click(Sender: TObject);

Var

MessageBoxCaption: String;

MessageBoxText: String;

IsMessageShow: Boolean;

Begin

IsMessageShow := False;

If Not IsMessageShow Then

Begin

IsMessageShow := True;

MessageBoxCaption := 'О разработчике';

MessageBoxText := 'Машевский Даниил Витальевич, группа 351003, Лабораторная №

7.2';

MessageBox(Handle, PChar(MessageBoxText), PChar(MessageBoxCaption), MB\_OK);

End;

IsMessageShow := False;

End;

Procedure TLaba72Form.NOpenFileClick(Sender: TObject);

Var

OpenDialog: TOpenDialog;

FileLines: TStrings;

VertexCount, VertexAValue, VertexBValue: Integer;

I, J: Integer;

Begin

OpenDialog := TOpenDialog.Create(Self);

OpenDialog.Filter := 'Text Files|\*.txt';

If OpenDialog.Execute Then

Begin

FileLines := TStringList.Create;

Try

FileLines.LoadFromFile(OpenDialog.FileName);

//Считываем количество вершин

If Not TryStrToInt(FileLines[0], VertexCount) Then

Begin

ShowMessage('Первая строка файла должна содержать количество вершин в

графе.');

Exit;

End;

SetLength(AdjacencyMatrix, VertexCount, VertexCount);

For I := 1 To FileLines.Count - 1 Do //Начинаем считывать с первой строки

после количества вершин

Begin

If Not TryStrToInt(Copy(FileLines[I], 1, Pos(' ', FileLines[I]) - 1), VertexAValue) Or

Not TryStrToInt(Copy(FileLines[I], Pos(' ', FileLines[I]) + 1, Length(FileLines[I])), VertexBValue) Then

Begin

ShowMessage('Некорректные данные в строке ' + IntToStr(I));

Exit;

End;

AdjacencyMatrix[VertexAValue, VertexBValue] := 1;

AdjacencyMatrix[VertexBValue, VertexAValue] := 1;

End;

//Отображение матрицы смежности в StringGrid

MatrixSmezhStringGrid.RowCount := VertexCount + 1;

MatrixSmezhStringGrid.ColCount := VertexCount + 1;

For I := 0 To VertexCount - 1 Do

Begin

MatrixSmezhStringGrid.Cells[I + 1, 0] := IntToStr(I);

MatrixSmezhStringGrid.Cells[0, I + 1] := IntToStr(I);

For J := I + 1 To VertexCount - 1 Do

Begin

MatrixSmezhStringGrid.Cells[J + 1, I + 1] := IntToStr(AdjacencyMatrix[I, J]);

MatrixSmezhStringGrid.Cells[I + 1, J + 1] := IntToStr(AdjacencyMatrix[J, I]);

End;

End;

Finally

FileLines.Free;

End;

End;

OpenDialog.Free;

MainMenu.Items[0].Items[1].Enabled := True;

End;

Procedure WriteStrIntoFile(Path: String; Massiv: TArr);

Var

F: TextFile;

I, J: Integer;

Mes: String;

Begin

Mes := 'Результат вычислений:'#13#10;

AssignFile(F, Path);

Rewrite(F);

Write(F, Mes);

For I := 0 To High(Massiv) Do

Begin

For J := 0 To High(Massiv) Do

Begin

Write(F, Massiv[I][J], ' ');

End;

WriteLn(F); //Переход на новую строку после каждой строки матрицы

End;

CloseFile(F);

End;

Procedure TLaba72Form.NSaveFileClick(Sender: TObject);

Var

SaveDialog: TSaveDialog;

MText, S1, S2: String;

FileName, Path: String;

MessageBoxCaption: String;

MessageBoxText: String;

IsMessageShow: Boolean;

InputFile: TextFile;

I, J, ArrLen: Integer;

Begin

SaveDialog := TSaveDialog.Create(Nil);

SaveDialog.Title := 'Сохранить файл';

SaveDialog.Filter := 'Текстовые файлы (\*.txt)|\*.txt';

SaveDialog.DefaultExt := 'txt';

If SaveDialog.Execute Then

Begin

FileName := SaveDialog.FileName;

If Not FileName.EndsWith('.txt') Then

FileName := FileName + '.txt';

Path := FileName;

WriteStrIntoFile(Path, AdjacencyMatrix);

SaveDialog.Free;

IsMessageShow := False;

If Not IsMessageShow Then

Begin

IsMessageShow := True;

MessageBoxCaption := 'Работа с файлом:';

MessageBoxText := 'Запись в файл прошла успешно!';

MessageBox(Handle, PChar(MessageBoxText), PChar(MessageBoxCaption), MB\_OK);

End;

IsMessageShow := False;

End;

End

End.

**Код программы на Java:**

import java.util.ArrayList;

import java.util.Scanner;

import java.io.File;

import java.io.FileNotFoundException;

import java.util.List;

public class Graph {

private static List<Integer> vertices = new ArrayList<>();

private static boolean isNumberInArray(int num) {

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

if (vertices.get(i) == num) {

return true;

}

}

return false;

}

private static int getIndex(int num) {

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

if (vertices.get(i) == num) {

return i;

}

}

return -1;

}

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int len = 0, conf;

System.out.println("Enter 1 to read data from a file, 0 - to read from

console");

conf = scanner.nextInt();

String path;

int temp;

List<Integer> A = new ArrayList<>();

List<Integer> B = new ArrayList<>();

if (conf == 1) {

System.out.println("Enter the file path");

path = scanner.next();

try {

File file = new File(path);

Scanner fileScanner = new Scanner(file);

len = fileScanner.nextInt();

for (int i = 0; i < len; ++i) {

temp = fileScanner.nextInt();

A.add(temp);

temp = fileScanner.nextInt();

B.add(temp);

}

fileScanner.close();

} catch (FileNotFoundException e) {

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

}

} else {

System.out.println("Enter the number of edges");

len = scanner.nextInt();

System.out.println("Enter the edges:");

for (int i = 0; i < len; ++i) {

temp = scanner.nextInt();

A.add(temp);

temp = scanner.nextInt();

B.add(temp);

}

}

for (int i = 0; i < len; ++i) {

if (!isNumberInArray(A.get(i)))

vertices.add(A.get(i));

if (!isNumberInArray(B.get(i)))

vertices.add(B.get(i));

}

// Create an adjacency matrix

int[][] matrix = new int[vertices.size()][vertices.size()];

for (int i = 0; i < len; ++i) {

int indexA = getIndex(A.get(i));

int indexB = getIndex(B.get(i));

matrix[indexA][indexB] = 1;

matrix[indexB][indexA] = 1;

}

// Print the adjacency matrix

for (int i = 0; i < matrix.length; ++i) {

for (int j = 0; j < matrix[i].length; ++j) {

System.out.print(matrix[i][j] + " ");

}

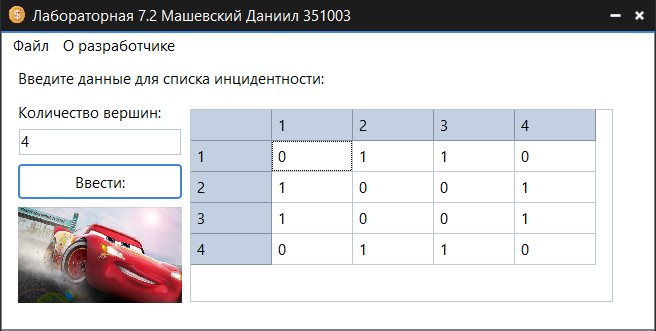
System.out.println();

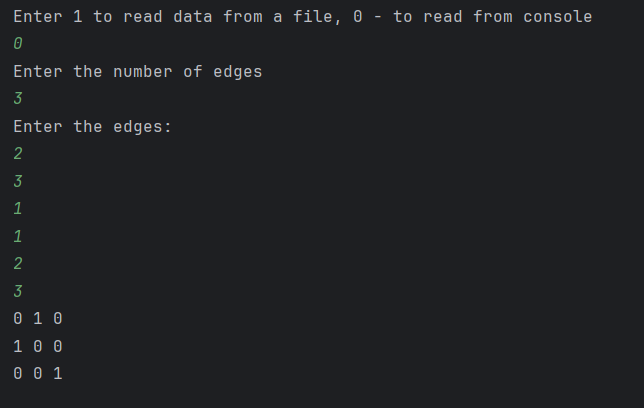
}

}

}

**Скриншоты:**





**Блок-схема:**



