**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

*Факультет інформатики та обчислювальної техніки*

*Кафедра обчислювальної техніки*

**Лабораторна робота №5**

*з дискретної математики*

*Виконав студент групи ІО-44*

***Барабаш Т.А.***

*Перевірив:*

***Новотарський М.А****.*

**2015**

**ЛАБОРАТОРНА РОБОТА №5**

*Розфарбовування графа, алгоритми розфарбування*

**Мета:** вивчення способів правильного розфарбовування графа.

**I. Завдання**

**Варіант 3. Розфарбування графа за модифікованим евристичним алгоритмом**

А) Виконати завдання 3 до лабораторної роботи: Вивчити основні означення та теореми про розфарбування графів. Створити програму розфарбування графів яка реалізує модифікований евристичний алгоритм розфарбування.

Б) Программа повинна дозволяти розфарбування довільного.B) Вивести у графічному режимі розфарбований граф.

**ІI. Короткі теоретичні відомості**

**МОДИФІКОВАНИЙ ЕВРИСТИЧНИЙ АЛГОРИТМ РОЗФАРБУВАННЯ**

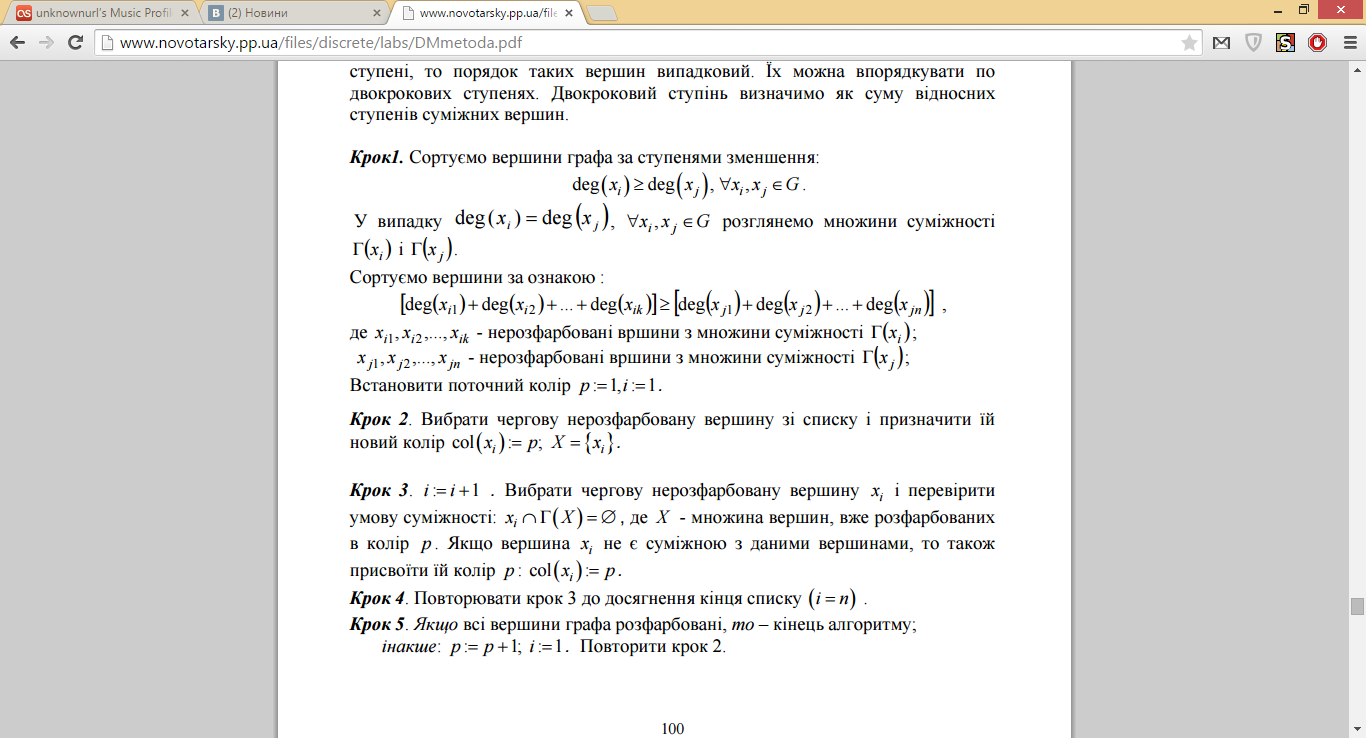
**Попередні визначення**

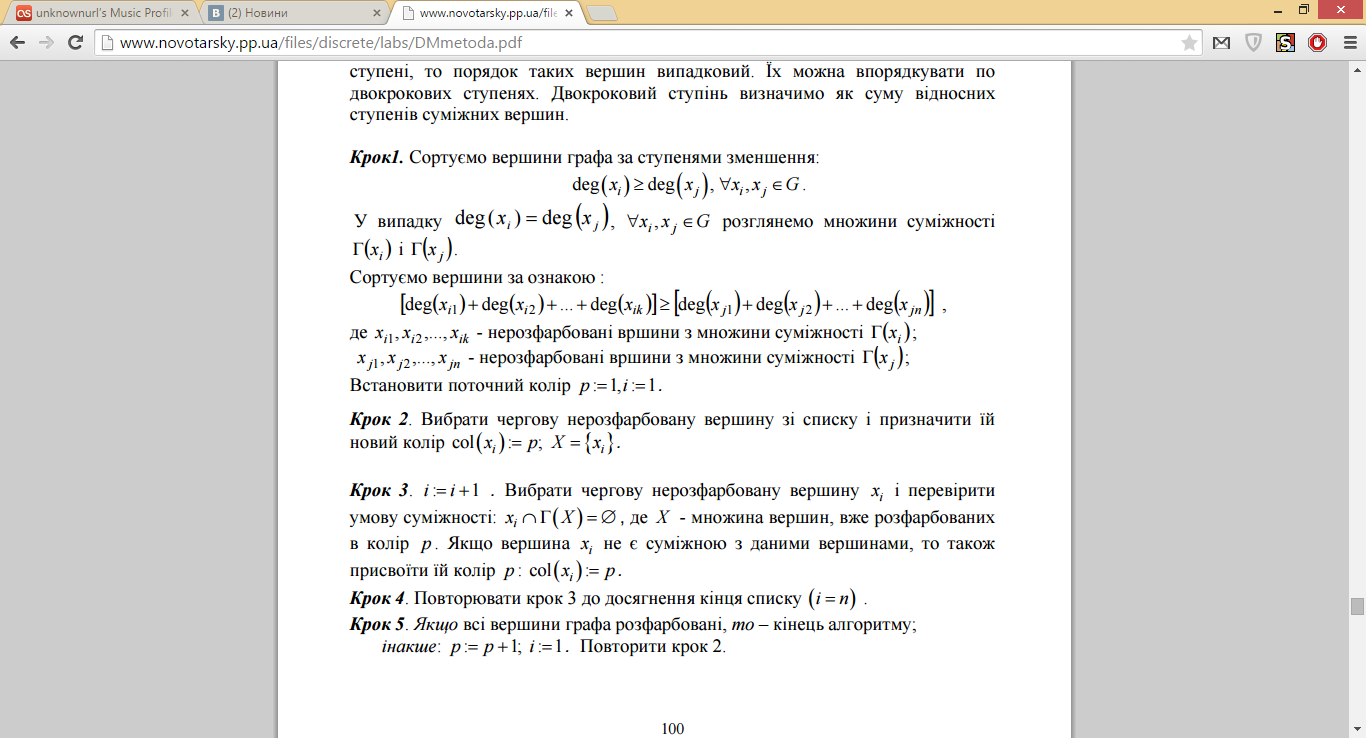
**Визначення 1.** Відносний ступінь – це ступінь нерозфарбованих вершин у нерозфарбованому підграфі даного графа.

**Визначення 2.** Двокроковий відносний ступінь – сума відносних ступенів суміжних вершин у нерозфарбованому підграфі.

Проста модифікація описаної вище евристичної процедури базується на переупорядкуванні нерозфарбованих вершин по незростанню їх відносних ступенів.

Дана модифікація полягає у тому, що якщо дві вершини мають однакові ступені, то порядок таких вершин випадковий. Їх можна впорядкувати по двокрокових ступенях. Двокроковий ступінь визначимо як суму відносних ступенів суміжних вершин.





Даний алгоритм від попереднього відрізняється ускладненням процедури сортування SortNodes, яка при сортуванні вершин з однаковими ступенями враховує двукроковий ступінь.

**IIІ. Код програми**

unit Unit2;

{$mode objfpc}{$H+}

interface

uses

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

StdCtrls, Grids,Unit3,Unit5;

Const Nmax=100; {\*максимальна кількість вершин графа\*}

type

{ TOperForm }

TOperForm = class(TForm)

Download: TButton;

Label1: TLabel;

Painting: TButton;

ExitButton: TButton;

GrImage: TImage;

ResultGrid: TStringGrid;

procedure DownloadClick(Sender: TObject);

procedure PaintingClick(Sender: TObject);

procedure ExitButtonClick(Sender: TObject);

procedure ColloringTopOf(number: integer; colorOf: integer);

procedure PaintGraph;

procedure DegForming;

procedure SortNodes;

private

{ private declarations }

public

{ public declarations }

end;

TArr = Array [1..Nmax] of Integer;

TA = Array [1..Nmax, 1..Nmax] of Byte;

TBoolean = Array[1..Nmax] of boolean;

var

OperForm: TOperForm;

Matr: TA; {\*матриця суміжності графа\*}

ColArr: TArr; {\*масив номерів фарб для кожної вершини графа\*}

DegArr: TArr; {\*масив ступенів вершин\*}

SortArr:TArr;{\*відсортований за зменшенням ступенів масив вершин\*}

CurCol: Byte; {\*поточний номер фарби\*}

SortUse:TBoolean;

implementation

{$R \*.lfm}

{ TOperForm }

procedure TOperForm.ExitButtonClick(Sender: TObject);

begin

Close;

end;

procedure TOperForm.PaintGraph;

var i,j:Integer;

h,x,y,x0,y0,x1,y1:Integer;

Angle,a:Extended;

begin

GrImage.Canvas.Pen.Width:= 1;

GrImage.Canvas.Pen.Color:=clBlack;

GrImage.Canvas.Brush.Color := clwhite;

GrImage.Canvas.Rectangle(0, 0, 600, 600);

GrImage.Canvas.Font.Size:=10;

GrImage.Canvas.Pen.Color:=clBlack;

X0:=300;

Y0:=300;

a:=0;

h:=1;

GrImage.Canvas.Pen.Width:= 2;

GrImage.Canvas.Pen.Color:=clBlack;

While a<360 do

begin

CoordArr[h].X:=x0+Round( 220 \* cos(a\*2\*pi/360));

CoordArr[h].Y:=x0-Round( 220 \* sin(a\*2\*pi/360));

GrImage.Canvas.MoveTo(CoordArr[h].X,CoordArr[h].Y);

GrImage.Canvas.Ellipse(CoordArr[h].X-6,CoordArr[h].Y-6,CoordArr[h].X+6,CoordArr[h].Y+6);

If (a>45 ) AND (a<225) then

begin

x:=x0+Round( (220+30) \* cos(a\*2\*pi/360));

y:=x0-Round( (220+30) \* sin(a\*2\*pi/360));

end

else

begin

x:=x0+Round( (220+15) \* cos(a\*2\*pi/360));

y:=x0-Round( (220+15) \* sin(a\*2\*pi/360));

end;

GrImage.Canvas.TextOut(x-2,y-2,IntToStr(h));

Inc(h);

a:=a+360/NumNodes;

end;

For i:=1 to NumNodes do

For j:=1 to NumNodes do

begin

If ResultGrid.Cells[j,i]='1' then

begin

If i<>j then

begin

GrImage.Canvas.MoveTo(CoordArr[i].X,CoordArr[i].Y);

GrImage.Canvas.LineTo(CoordArr[j].X,CoordArr[j].Y);

end else

begin

Angle:=(i-1)\*360/NumNodes;

x1:=CoordArr[i].X+Round( 25 \* cos(Angle\*2\*pi/360));

y1:=CoordArr[i].Y-Round( 25 \* sin(Angle\*2\*pi/360));

GrImage.Canvas.Ellipse(x1-25,y1-25,x1+25,y1+25);

end;

end;

end;

end;

procedure TOperForm.DownloadClick(Sender: TObject);

var F : textfile;

Str:String;

m,i,j,x,y,x0,y0,x1,y1,h:Integer;

Angle,a:Extended;

begin

AssignFile(F,'DATA\P3.TXT');

Reset(F);

Readln(F, Str);

ResultGrid.RowCount:=StrToInt(str);

Readln(F, Str);

ResultGrid.ColCount:=StrToInt(str);

NumNodes:=ResultGrid.ColCount-1;

For i:=1 to ResultGrid.RowCount-1 do

For j:=1 to ResultGrid.ColCount-1 do

begin

Readln(F,str);

if(str = '1') then

ResultGrid.Cells[j,i]:='1'

else

ResultGrid.Cells[j,i]:='0';

end;

CloseFile(F);

For i:=1 to ResultGrid.RowCount-1 do

For j:=1 to ResultGrid.ColCount-1 do

begin

if (ResultGrid.Cells[j,i]='0') and (ResultGrid.Cells[i,j]='1') then

ResultGrid.Cells[j,i]:='1';

end;

For i:=1 to NumNodes do

begin

ResultGrid.Cells[i,0]:=IntToStr(i);

ResultGrid.Cells[0,i]:=IntToStr(i);

end;

For i:=1 to ResultGrid.RowCount-1 do

For j:=1 to ResultGrid.ColCount-1 do

Matr[j,i]:=strtoint(ResultGrid.Cells[i,j]);

PaintGraph;

end;

procedure TOperForm.PaintingClick(Sender: TObject);

var i,j,n : integer;

flag:boolean;

begin

CurCol:=1;

DegForming; {\*Формування масиву ступенів вершин\*}

SortNodes; {\*Формування массиву відсортованих вершин SortArr\*}

For n:=1 to NumNodes do

begin

If ColArr[SortArr[n]]=0 then

begin

ColArr[SortArr[n]]:=CurCol;

For j:=n+1 to NumNodes do

begin

if (ColArr[SortArr[j]]=0) and (Matr[SortArr[j],SortArr[n]]=0) then

begin

flag:=true;

for i:=n+1 to j do

if (Matr[SortArr[i],SortArr[j]]=1) and (ColArr[SortArr[i]]=CurCol)then flag:=false;

if flag = true then ColArr[SortArr[j]]:=CurCol;

end;

end;

Inc(CurCol);

end;

end;

For i:=1 to NumNodes do

ColloringTopOf(i,ColArr[i]);

end;

Procedure TOperForm.DegForming; {\*Процедура формування масиву ступенів вершин\*}

Var i,j,n:Byte;

Function DegCount(m:Byte):Integer;

Var k, Deg:Integer;

Begin

Deg:=0;

For k:=1 to NumNodes do Deg:= Deg+Matr[k,m];

DegCount:=Deg;

End;

Begin

For j:=1 to NumNodes do

begin

ColArr[j]:=0;

DegArr[j]:= DegCount(j)\*100;

For i:=1 to NumNodes do

If (Matr[j,i]=1 )then DegArr[j]:= DegArr[j]+DegCount(i);

end;

End;

Procedure TOperForm.SortNodes; {\*Сортування вершин за ступенями\*}

Var c,k,i,j:Byte; max:integer;

Begin

for k:=1 to numnodes do

SortUse[k]:=false;

For k:=1 to NumNodes do

begin

i:=1;

while ( SortUse[i]=true)and(i<(NumNodes)) do

i:=i+1;

max:=DegArr[i];

c:=i;

for j:=i to NumNodes do

if (SortUse[j]=false)and(DegArr[j]>max) then

begin

max:=DegArr[j];

c:=j;

end;

SortUse[c]:=true;

SortArr[k]:=c;

end;

End;

procedure TOperForm.ColloringTopOf(number: integer; colorOf: integer);

var i:Integer;

x,y: integer;

x0,y0: integer;

a: Extended;

h: integer;

begin

case colorOf of

1: GrImage.Canvas.Pen.Color:=clRed;

2: GrImage.Canvas.Pen.Color:=clBlue;

3: GrImage.Canvas.Pen.Color:=clLime;

4: GrImage.Canvas.Pen.Color:=clYellow;

5: GrImage.Canvas.Pen.Color:=clAqua;

6: GrImage.Canvas.Pen.Color:=clOlive;

7: GrImage.Canvas.Pen.Color:=clTeal;

8: GrImage.Canvas.Pen.Color:=clGreen;

9: GrImage.Canvas.Pen.Color:=clMaroon;

10: GrImage.Canvas.Pen.Color:=clFuchsia;

11: GrImage.Canvas.Pen.Color:=clGray;

end;

X0:=300;

Y0:=300;

a:=0;

h:=1;

GrImage.Canvas.Pen.Width:= 9;

While a<360 do

begin

CoordArr[h].X:=x0+Round( 220 \* cos(a\*2\*pi/360));

CoordArr[h].Y:=x0-Round( 220 \* sin(a\*2\*pi/360));

if number = h then

begin

GrImage.Canvas.MoveTo(CoordArr[h].X,CoordArr[h].Y);

GrImage.Canvas.Ellipse(CoordArr[h].X-1,CoordArr[h].

Y-1,CoordArr[h].X+1,CoordArr[h].Y+1);

end;

If (a>45 ) AND (a<225) then

begin

x:=x0+Round( (220+30) \* cos(a\*2\*pi/360));

y:=x0-Round( (220+30) \* sin(a\*2\*pi/360));

end else

begin

x:=x0+Round( (220+15) \* cos(a\*2\*pi/360));

y:=x0-Round( (220+15) \* sin(a\*2\*pi/360));

end;

GrImage.Canvas.TextOut(x-2,y-2,IntToStr(h));

If (a>45 ) AND (a<225) then

begin

x:=x0+Round( (240+30) \* cos(a\*2\*pi/360));

y:=x0-Round( (240+30) \* sin(a\*2\*pi/360));

end else

begin

x:=x0+Round( (240+15) \* cos(a\*2\*pi/360));

y:=x0-Round( (240+15) \* sin(a\*2\*pi/360));

end;

GrImage.Canvas.TextOut(x-2,y-2,IntToStr(DegArr[h] div 100));

If (a>45 ) AND (a<225) then

begin

x:=x0+Round( (260+30) \* cos(a\*2\*pi/360));

y:=x0-Round( (260+30) \* sin(a\*2\*pi/360));

end else

begin

x:=x0+Round( (260+15) \* cos(a\*2\*pi/360));

y:=x0-Round( (260+15) \* sin(a\*2\*pi/360));

end;

GrImage.Canvas.TextOut(x-2,y-2,IntToStr(DegArr[h] mod 100));

Inc(h);

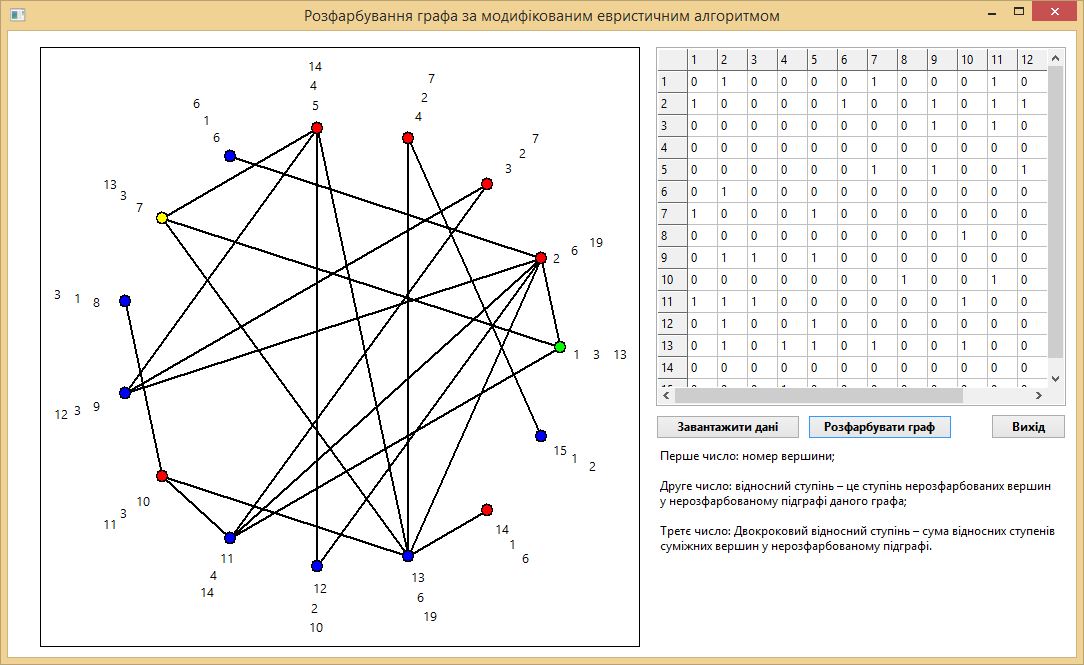
a:=a+360/NumNodes;

end;

end;

end.

**IV. Результат**



**V. Висновок та аналіз результатів**

У ході лабораторної роботи було закріплено знання з теорії графів. При виконанні роботи граф було розфарбовано за допомогою модифікованого евристичного методу. Для цього методу було створено просту програму з візуальною частиною. Отримані результати підтверджують правильність роботи алгоритму.