МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Лабораторна робота №4

з дисципліни «Дискретна математика»

на тему

«Графи. Способи представлення графів. Остовні дерева. Пошук

найкоротших шляхів»

*підготував студент*

*групи ІО-31*

*Головко Олег Сергійович*

Варіант завдання №4:

«Алгоритм Дейкстри пошуку найкоротшого шляху в графі»

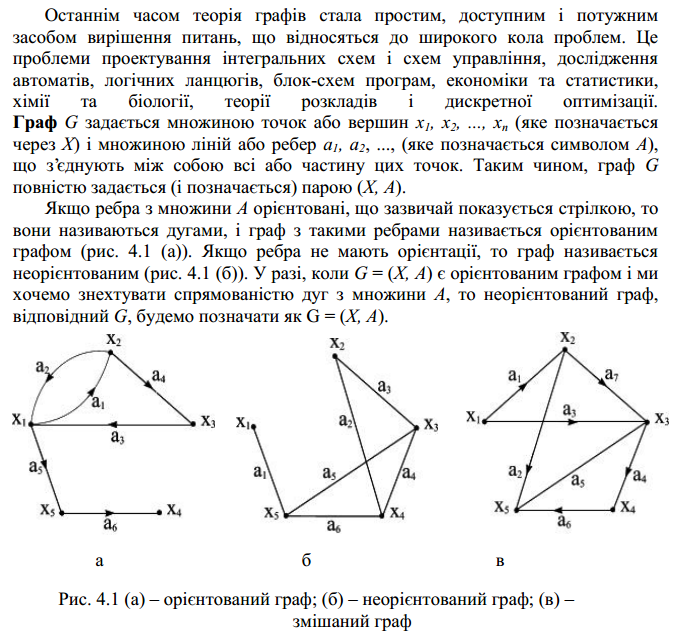
**м. Київ**

**2014 р.**

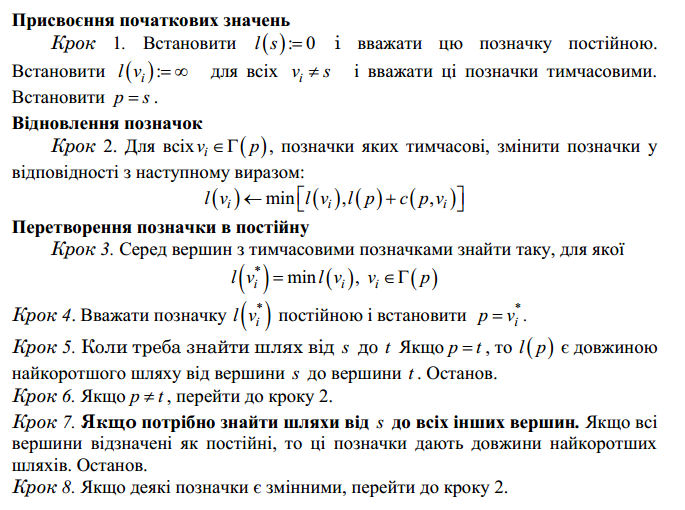
**Мета роботи:** Вивчення властивостей графів, способів їх представлення та основних алгоритмів на графах.

**Завдання :** створити програму, яка реалізує один з алгоритмів на графах.

**Теоретичні основи:**



**Теоретичний опис алгоритму Дейкстри**

****

**Код програми:**

procedure TOperForm.InputClick ( Sender: TObject ) ; //input the info

var

myfile1:TextFile;

str\_tmp1: String;

index\_tmp, i : integer;

begin

AssignFile(myfile1,'DATA\P4.TXT');

Reset(myfile1);

index\_tmp:=-1;

readln( myfile1, str\_tmp1 );

readln( myfile1, str\_tmp1 );

while not eof(myfile1) do

begin

index\_tmp:=index\_tmp+1;

readln( myfile1, str\_tmp1 );

MatrixGrid.Cells[ (index\_tmp mod (MatrixGrid.ColCount-1))+1,

(index\_tmp div (MatrixGrid.RowCount-1))+1 ]:= str\_tmp1;

end;

closefile(myfile1);

for i:=0 to MatrixGrid.RowCount-1 do

begin

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

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

end;

NodesPaint;

EdgesPaint;

end;

procedure TOperForm.endofChange ( Sender: TObject ) ; //enable of CertainDis

begin

if ( endof.Text <> '' ) and ( beginof.Text <> '' ) then

CertaindDis.Enabled := true

else

CertaindDis.Enabled := false;;

end;

procedure TOperForm.beginofChange ( Sender: TObject ) ; //enable of ALlDis

begin

if beginof.Text <> '' then

AllDis.Enabled := true

else

AllDis.Enabled := false;

end;

procedure TOperForm.AllDisClick ( Sender: TObject ) ;

var

count, index, min, start, u, i,finish, endOfArray : integer;

distance : array [ 1..15 ] of integer;

visited : array [ 1..15 ] of boolean;

steps : array [ 1..20 ] of String;

begin

endOfArray := MatrixGrid.RowCount-1;

start := StrToInt(beginof.Text); //the first node

for i := 1 to endOfArray do //first values of arrays

begin

distance[ i ] := maxint;

visited[ i ] := false;

end;

distance[ start ] := 0;

for count := 1 to endOfArray do //find the shortest way

begin

min := maxint;

for i := 1 to endOfArray do

begin

if ( not visited[ i ] ) and ( distance [ i ] <=min ) then

begin

min := distance[ i ];

index := i;

end;

end;

u := index;

visited[ u ] := true;

for i := 1 to endOfArray do

if ( not visited[ i ] ) and ( MatrixGrid.Cells[ u , i] <> IntToStr(0) ) and

( distance[ u ] <> maxint ) and

( distance[ u ] + StrToInt( MatrixGrid.Cells[ u , i ] ) < distance [ i ] ) then

distance [ i ] := distance[ u ] + StrToInt( MatrixGrid.Cells[ u , i ] ) ;

end;

//

for i := 1 to endOfArray do

if distance[ i ] <> maxint then

StringGrid1.Cells[ 0 , i-1] := IntToStr( start ) + ' > ' + IntToStr( i ) +

' = ' + IntToStr( distance[ i ] )

else

StringGrid1.Cells[ 0 , i-1] := IntToStr( start ) + ' > ' + IntToStr( i ) +

' = ' + 'the pass is unable ' ;

end;

procedure TOperForm.CertaindDisClick ( Sender: TObject ) ; //certain distance

var

count, index, min, start, u, i,finish, endOfArray,j : integer;

distance : array [ 1..15 ] of integer;

visited : array [ 1..15 ] of boolean;

steps : array [ 1..20 ] of integer;

begin

for i := 0 to StringGrid1.RowCount-1 do

StringGrid1.Cells[0,i] := '';

endOfArray := MatrixGrid.RowCount-1;

start := StrToInt(beginof.Text); //the first node

finish := StrToInt(endof.Text);

for i := 1 to endOfArray do //first values of arrays

begin

distance[ i ] := maxint;//StrToInt( MatrixGrid.Cells[i, start] );

visited[ i ] := false;

steps[i] := start;

end;

distance[ start ] := 0;

steps[start] := 0;

for count := 1 to endOfArray do //find the shortest way

begin

min := maxint;

for i := 1 to endOfArray do

begin

if ( not visited[ i ] ) and ( distance [ i ] <=min ) then

begin

min := distance[ i ];

index := i;

end;

end;

u := index;

visited[ u ] := true;

for i := 1 to endOfArray do

if ( not visited[ i ] ) and ( MatrixGrid.Cells[ u , i] <> IntToStr(0) ) and

( distance[ u ] <> maxint ) and

( distance[ u ] + StrToInt( MatrixGrid.Cells[ u , i ] ) < distance [ i ] ) then

begin

distance [ i ] := distance[ u ] + StrToInt( MatrixGrid.Cells[ u , i ] ) ;

steps[i]:=u;

end;

end;

j:=0;

GraphImage.Canvas.Pen.Width:= 4;

GraphImage.Canvas.Pen.Color:=clBlack;

GraphImage.picture:=nil ;

nodespaint;

finish:=steps[finish];

StringGrid1.Cells[0, 0] := (endof.Text);

GraphImage.Canvas.MoveTo(CoordArr[StrToInt(endof.text)].X,CoordArr[StrToInt(endof.text)].Y);

GraphImage.Canvas.LineTo(CoordArr[finish].X,CoordArr[finish].Y); //make edge

while finish <> 0 do

begin

GraphImage.Canvas.MoveTo(CoordArr[finish].X,CoordArr[finish].Y); //begining

StringGrid1.Cells[0,0] := StringGrid1.Cells[0,0] + ' <-' + IntToStr(finish);

finish:=steps[finish];

if finish <>0 then

GraphImage.Canvas.LineTo(CoordArr[finish].X,CoordArr[finish].Y)

end;

StringGrid1.Cells[0,0] := StringGrid1.Cells[0,0] + ' the distance is ' +

IntToStr( distance[StrToInt(endof.Text)]) ;

end;

procedure TOperForm.ClearClick ( Sender: TObject ) ; //clear

Var

i : integer;

begin

endof.Text:= '';

beginof.Text:='';

for i := 0 to StringGrid1.RowCount-1 do

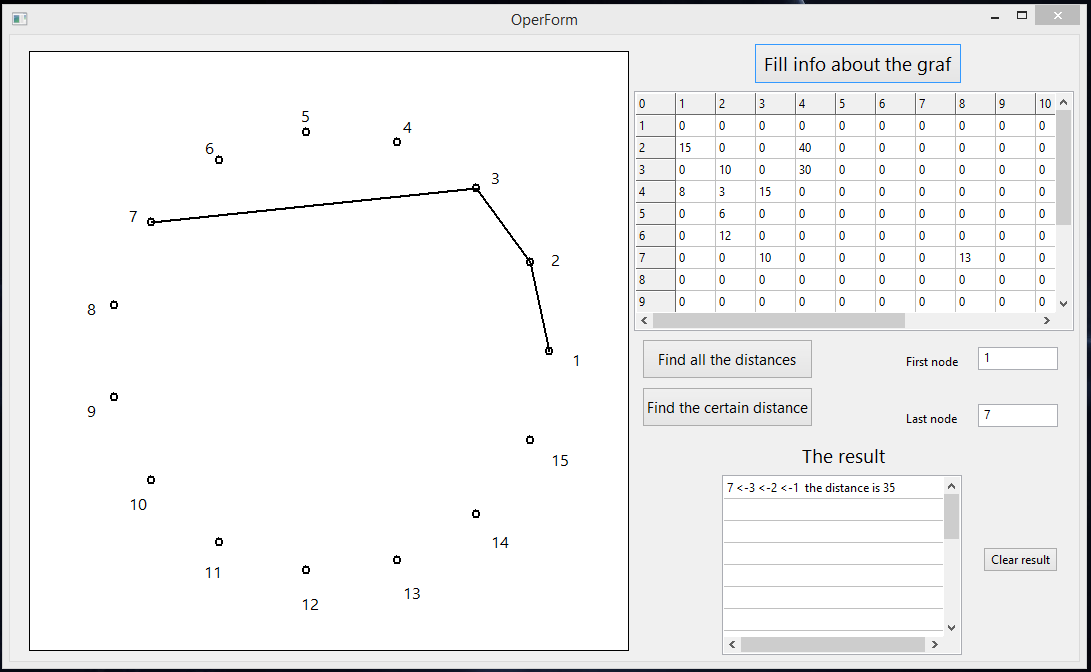
StringGrid1.Cells[0,i] := '';

NodesPaint;

EdgesPaint;

end;

**Результат роботи програми:**



**Висновки:**

Я реалізував програму, яка шукає найкоротші відстані між будь-якими 2-ма вершинами графа на основі алгоритму Дейкстри. Також можливий пошук найкоротших шляхів з заданої вершини до усіх інших.