Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра програмування та захисту інформації

**Звіт**

з виконаної лабораторної роботи № 7

дисципліни “ Дискретна математика ”

на тему

«Алгоритми на скінчених графах»

Виконав :

студент академічної групи КІ-15

Аннаєв А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив :

Викладач

Гермак В.С.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кропивницкий- 2017

**Лабораторная работа №7**

**ТЕМА:** Линейный синтез конечных графов

**ЦЕЛЬ:** Получить навыки линейного синтеза дискретная объектов и анализа наследования свойств.

Задание :

Выполнить построение множеств фундаментальных циклов и множеств простых циклов графа из приложения 1 номер которого совпадает с номером вашей фамилии в списке группы, как вручную, так и с помощью программного средства методом в глубину графа.

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Wi2

{

class Program

{

static List<string> catalogCycles;

static List<Vertex> V;

static List<Edge> E;

static void Main(string[] args)

{

catalogCycles = new List<string>();

V = new List<Vertex>();

E = new List<Edge>();

//добавить вершины и рёбра

V.Add(new Vertex(1, 1));

E.Add(new Edge(1, 2));

V.Add(new Vertex(2, 1));

E.Add(new Edge(3, 4));

V.Add(new Vertex(2, 3));

E.Add(new Edge(4, 5));

V.Add(new Vertex(1, 4));

E.Add(new Edge(3, 4));

V.Add(new Vertex(1, 3));

E.Add(new Edge(2, 3));

cyclesSearch();

System.Console.Read();

}

private static void cyclesSearch()

{

int[] color = new int[V.Count];

for (int i = 0; i < V.Count; i++)

{

for (int k = 0; k < V.Count; k++)

color[k] = 1;

List<int> cycle = new List<int>();

cycle.Add(i + 1); System.Console.WriteLine(i);

DFScycle(i, i, E, color, -1, cycle);

}

}

private static void DFScycle(int u, int endV, List<Edge> E, int[] color, int unavailableEdge, List<int> cycle)

{

//если u == endV, то эту вершину перекрашивать не нужно, иначе мы в нее не вернемся, а вернуться необходимо

if (u != endV)

color[u - 1] = 2;

else if (cycle.Count >= 2)

{

cycle.Reverse();

string s = cycle[0].ToString();

for (int i = 1; i < cycle.Count; i++)

s += "-" + cycle[i].ToString();

bool flag = false; //есть ли палиндром для этого цикла графа в List<string> catalogCycles?

for (int i = 0; i < catalogCycles.Count; i++)

if (catalogCycles[i].ToString() == s)

{

flag = true;

break;

}

if (!flag)

{

cycle.Reverse();

s = cycle[0].ToString();

for (int i = 1; i < cycle.Count; i++)

s += "-" + cycle[i].ToString();

catalogCycles.Add(s);

}

return;

}

for (int w = 0; w < E.Count; w++)

{

if (w == unavailableEdge)

continue;

if (color[E[w].v2 - 1] == 1 && (E[w].v1 - 1) == u)

{

List<int> cycleNEW = new List<int>(cycle);

cycleNEW.Add(E[w].v2 + 1);

System.Console.Write(cycleNEW[E[w].v2 - 1] + "-");

DFScycle(E[w].v2, endV, E, color, w, cycleNEW);

color[E[w].v2 - 1] = 1;

}

else if (color[E[w].v1] == 1 && E[w].v2 == u)

{

List<int> cycleNEW = new List<int>(cycle);

cycleNEW.Add(E[w].v1 + 1);

DFScycle(E[w].v1, endV, E, color, w, cycleNEW);

color[E[w].v1] = 1;

}

}

}

}

//вершина

class Vertex

{

public int x, y;

public Vertex(int x, int y)

{

this.x = x;

this.y = y;

}

}

//ребро

class Edge

{

public int v1, v2;

public Edge(int v1, int v2)

{

this.v1 = v1;

this.v2 = v2;

}

}

}

