**Kauno technologijos universitetas**

Informatikos fakultetas

Taikomosios informatikos katedra

**Diskrečiosios struktūros (p170b008)**

**Kursinis darbas**

**Užduoties Nr. B03**

Atliko: IFIN-8/3 gr. studentas *Dominykas Adomaitis*

Priėmė: dėst. *Audrius Nečiūnas*

KAUNAS

2019

# 1. Užduotis (Nr. B03)

Nustatyti, ar grafas yra dvidalis, ir jei taip, tai išskirti abi grafo puses.

# 2. Užduoties analizė

Grafas, kurio viršūnių aibę galima išskaidyti į du poaibius A ir B taip, kad kiekvienos briaunos galai priklausytų skirtingiems poaibiams, vadinamas dvidaliu grafu. Programai davus grafo viršūnių matrica programa duos atsakymą ar grafas dvidalis ar ne.

Iš pradžių kiekvienai grafo viršūnei priskiriama spalva (-1) ir pradinei viršūnei priskiriama spalva 1. Toliau tikrinant ar egzistuoja briauna iki kitų viršūnių ir ar tos viršūnės nenuspalvintos ta pačia spalva, jeigu sąlyga tinka viršūnė nuspalvinama kita priešinga spalva(0). Ciklai vykdomi kol visos viršūnės nuspalvinamos dviem skirtingomis spalvomis arba sąlyga nebeįvykdoma.

# 3. Programos tekstas

## Program.cs

using System;

using System.Collections;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Dvidalis\_grafas

{

class Matrica

{

const int CMaxEil = 1000; // didžiausias galimas eilučių skaičius

const int CMaxSt = 1000; // didžiausias galimas stulpelių skaičius

private double[,] A; // duomenų matrica

public int n { get; set; } // eilučių skaičius

public int m { get; set; } // stulpelių skaičiu

//----------------------------------------------------

/\*\* Pradinių matricos duomenų nustatymas \*/

//----------------------------------------------------

public Matrica()

{

n = 0;

m = 0;

A = new double[CMaxEil, CMaxSt];

}

//----------------------------------------------------

/\*\* Priskiria klasės matricos kintamajam reikšmę.

@param i - eilutės indeksas

@param j - stulpelio indeksas

@param reiksme - skaičius \*/

//----------------------------------------------------

public void Deti(int i, int j, double reiksme)

{

A[i, j] = reiksme;

}

//----------------------------------------------------

/\*\* Grąžina palukana.

@param i - eilutės indeksas

@param j - stulpelio indeksas \*/

//----------------------------------------------------

public double ImtiReiksme(int i, int j)

{

return A[i, j];

}

/\*\* Grąžina kiekį \*/

public int Imti() { return n; }

}

class Program

{

const string CFd = "..\\..\\grafas4.txt";

const string CFr = "..\\..\\rez.txt";

static void Main(string[] args)

{

if (File.Exists(CFr))

File.Delete(CFr);

Matrica matrica = new Matrica();

SkaitytiMatrica(ref matrica, CFd);

SpausdintiMatrica(matrica, CFr, "Grafo matrica");

int V = matrica.n;

int[] spalva = new int[V];

bool dvidalis = false;

dvidalis = isBipartite(matrica, 1, ref spalva);

Console.Write("Ar grafas dvidalis: ");

if (dvidalis == true)

Console.WriteLine("true");

else

Console.WriteLine("false");

FormuotiGrafoKoda(matrica);

SpausdintiVirsunes(matrica, spalva, dvidalis);

}

static bool isBipartite(Matrica matrica, int src, ref int[] spalva)

{

int V = matrica.n;

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

spalva[i] = -1;

//Priskia spalva pradinei virsunei

spalva[src] = 1;

//Sukuriama eile

Queue qt = new Queue();

qt.Enqueue(src);

//Kol eileje yra virsuniu

while (qt.Count != 0)

{

object pirmas = qt.Peek();

int u = int.Parse(string.Format("{0}", pirmas));

qt.Dequeue();

if (matrica.ImtiReiksme(u, u) == 1)

return false;

//Randa visas salia esancias nenuspalvintas virsunes

for (int v = 0; v < V; v++)

{

//Jeigu egzistuoja briauna nuo u iki v ir v virsune nenuspalvinta

if (matrica.ImtiReiksme(u, v) == 1 && spalva[v] == -1)

{

//Priskiria kita spalva nei u

spalva[v] = 1 - spalva[u];

qt.Enqueue(v);

}

//Jeigu egzistuoja briauna nuo u iki v ir v virsune nuspalvinta tokia pacia spalva kaip u

else if (matrica.ImtiReiksme(u, v) == 1 && spalva[v] == spalva[u])

return false;

}

}

//Visos virsunes nuspalvinamos 2 skirtingomis spalvomis ir grafas yra dvidalis

return true;

}

static void SpausdintiVirsunes(Matrica matrica, int[] spalva, bool dvidalis)

{

int x1 = 0;

int x2 = 0;

int V = matrica.n;

int[] spalva1 = new int[V];

int[] spalva2 = new int[V];

int skaitiklis = 10;

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

{

//Console.WriteLine(spalva[i]);

if(spalva[i] == -1)

{

if (skaitiklis == 10)

{

spalva[i] = 0;

skaitiklis = 9;

}

else if (skaitiklis == 9)

{

spalva[i] = 1;

skaitiklis = 10;

}

}

}

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

{

if (dvidalis == true)

{

if (spalva[i] == 1)

{

spalva1[x1] = i;

x1++;

}

else if (spalva[i] == 0)

{

spalva2[x2] = i;

x2++;

}

}

else

{

Console.WriteLine("Grafas ne dvidalis viršūnes nenuspalvintos 2 spalvomis");

break;

}

}

if (dvidalis == true)

{

Console.WriteLine("Pirmos spalvos viršūnės");

for (int i = 0; i < x1; i++)

{

Console.WriteLine(spalva1[i] + 1);

}

Console.WriteLine("Antros spalvos viršūnės");

for (int i = 0; i < x2; i++)

{

Console.WriteLine(spalva2[i] + 1);

}

}

}

static void SkaitytiMatrica(ref Matrica matrica, string fv)

{

int nn, mm;

double skaic;

string line;

using (StreamReader reader = new StreamReader(fv))

{

//nn = int.Parse(reader.ReadLine());

nn = 5;

mm = 5;

string[] parts;

matrica.n = nn;

matrica.m = mm;

for (int i = 0; i < nn; i++)

{

line = reader.ReadLine();

parts = line.Split(';');

for (int j = 0; j < mm; j++)

{

skaic = double.Parse(parts[j]);

matrica.Deti(i, j, skaic);

}

}

}

}

static void SpausdintiMatrica(Matrica matrica, string fv, string komentaras)

{

using (var fr = File.AppendText(fv))

{

Console.WriteLine(komentaras);

for (int i = 0; i < matrica.n; i++)

{

for (int j = 0; j < matrica.m; j++)

Console.Write("{0,4} ", matrica.ImtiReiksme(i, j));

Console.WriteLine();

}

}

}

static void FormuotiGrafoKoda(Matrica matrica)

{

Console.Write("U = {");

for(int i = 0; i < matrica.n; i++)

{

for (int j = 0; j < matrica.m; j++)

{

if (matrica.ImtiReiksme(i, j) == 1)

{

Console.Write(string.Format(" " +

"[{0} {1}]", i + 1, j + 1));

}

}

}

Console.WriteLine("};");

Console.Write("V = [");

for (int i = 0; i < matrica.n; i++)

{

Console.Write("{0} ", i + 1);

}

Console.WriteLine("];");

}

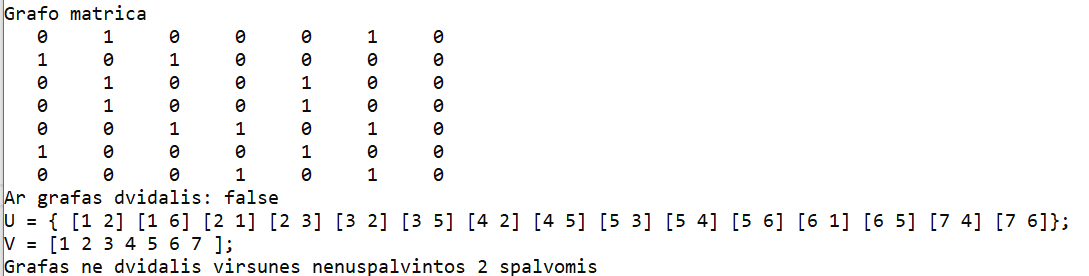
}

}

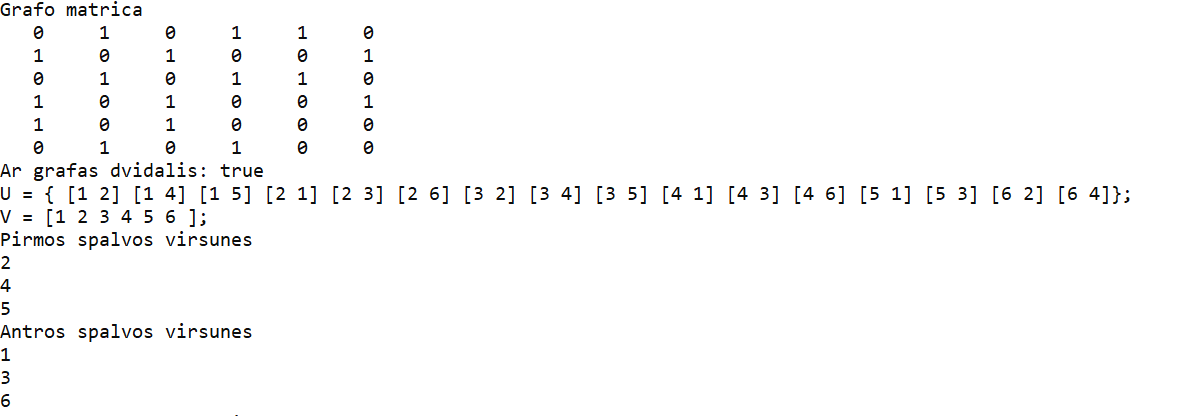
# 4. Testiniai pavyzdžiai

Buvo panaudoti trys testiniai pavyzdžiai.

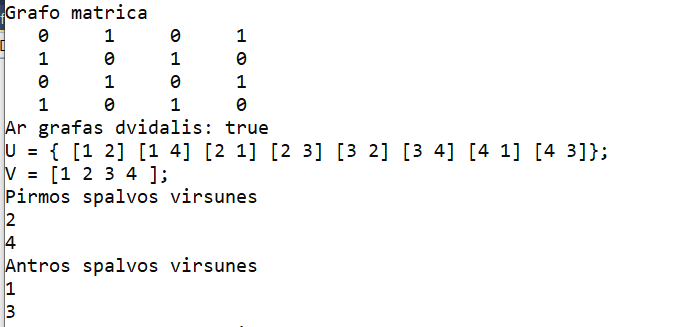
1 pav. Antrasis testinis variantas



2 pav. pirmasis teistinis variantas



3 pav. Trečiasis testinis variantas



# 5. Išvados

Sprendžiant iš pateiktų testinių pavyzdžių, programa veikia teisingai.

# 6. Literatūros sąrašas

1. „Diskrečiųjų struktūrų“ modulis „Moodle“ aplinkoje <https://moodle.ktu.edu/course/view.php?id=39>
2. <https://inf-knyga.nmakademija.lt/lt/latest/07_graf%C5%B3_pagrindai.html>
3. <https://en.wikipedia.org/wiki/Breadth-first_search>