Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №5

по курсу «Логика и основы алгоритмизации в ИЗ»

на тему: «Определение характеристик графов»

Выполнил:

студент группы 21ВВ1

Мишин Д.А.

Приняли:

Юрова О.В.

Акифьев И.В.

Пенза 2022

**Практическая часть**

**Задание**

1) Сгенерируйте (используя генератор случайных чисел) матрицу смежности для неориентированного графа G. Выведите матрицу на экран.

2) Определите размер графа G, используя матрицу смежности графа.

3) Найдите изолированные, концевые и доминирующие вершины.

**Листинг**

using System;

class Matrix

{

private int \_size;

private int[,] \_matrix;

private int \_countEdges;

public void Main()

{

\_size = Convert.ToInt32(Console.ReadLine());

\_matrix = GenerateMatrix(\_size);

PrintMatrix(\_matrix);

\_countEdges = FindNumberEges();

СonstructIncidenceMatrix();

Izolir(\_matrix, \_size);

Concev(\_matrix, \_size);

Domin(\_matrix, \_size);

Console.Read();

}

private int[,] GenerateMatrix(int size)

{

int[,] matrix = new int[size, size];

Random random = new Random();

for (int row = 0; row < size; row++)

{

for (int col = 0; col < row; col++)

{

matrix[row, col] = random.Next(0, 2);

matrix[col, row] = matrix[row, col];

}

}

return matrix;

}

private void PrintMatrix(int[,] matrix)

{

for (int row = 0; row < \_size; row++)

{

for (int col = 0; col < \_size; col++)

{

Console.Write($"{matrix[row, col]}, \t");

}

Console.WriteLine();

}

Console.WriteLine();

}

private int FindNumberEges()

{

int countEdges = 0;

for (int row = 0; row < \_size; row++)

{

for (int col = 0; col < row; col++)

{

if (\_matrix[row, col] == 1) countEdges++;

}

}

return countEdges;

}

private void СonstructIncidenceMatrix()

{

int[,] matrixIncidence = new int[\_countEdges, \_size];

int currentEdges = 0;

for (int row = 0; row < \_size; row++)

{

for (int col = 0; col < row; col++)

{

if (\_matrix[row, col] == 1)

{

matrixIncidence[currentEdges, col] = 1;

matrixIncidence[currentEdges, row] = 1;

if (\_countEdges - 1 != currentEdges) currentEdges++;

}

else

{

matrixIncidence[currentEdges, col] = 0;

matrixIncidence[currentEdges, row] = 0;

}

}

}

for (int i = 0; i < \_countEdges; i++)

{

for (int j = 0; j < \_size; j++)

{

Console.Write($"{matrixIncidence[i, j]}, \t");

}

Console.WriteLine();

}

Console.WriteLine();

}

private void Izolir(int[,] matrix, int size)

{

int vertexIz;

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

{

vertexIz = 0;

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

{

vertexIz = vertexIz + matrix[i, j];

}

if (vertexIz == 0)

{

Console.WriteLine("Вершина " + i + " является изолированной");

}

}

}

private void Concev(int[,] matrix, int size)

{

int vertexCon;

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

{

vertexCon = 0;

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

{

if (matrix[i, j] != 0)

{

vertexCon++;

}

}

if (vertexCon == 1)

{

Console.WriteLine("Вершина " + i + 1 + " является концевой");

}

}

}

private void Domin(int[,] matrix, int size)

{

int vertexDom;

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

{

vertexDom = 0;

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

{

if (matrix[i, j] != 0)

{

vertexDom++;

}

}

if (vertexDom == size - 1)

{

Console.WriteLine("Вершина " + i + " является доминирующей");

}

}

}

}

class Entrance

{

private static void Main(string[] args)

{

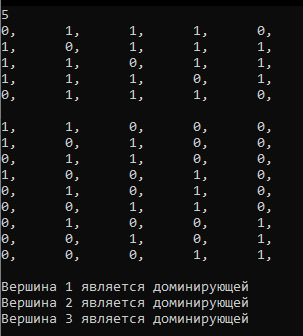
var laba = new Matrix();

laba.Main();

}

}

**Результат работы программы**



**Рисунок 2 — Результаты работы программы**

**Вывод:** в ходе данной работы я научился определять размер графа используя матрицу смежности, находить изолированные, концевые и доминирующие вершины.