Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №5

# по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «Определение характеристик графов»

Выполнили:

студенты группы 24ВВВ3

Агапов И.А.

Любченко В.К.

Приняли:

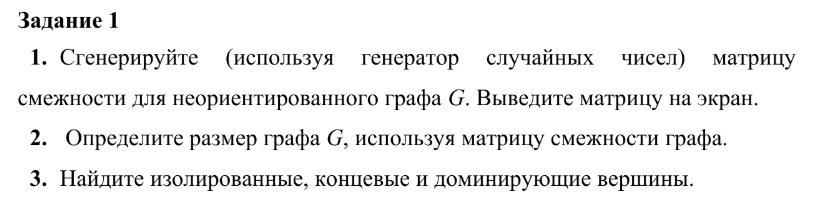
к.т.н., доцент Юрова О.В.

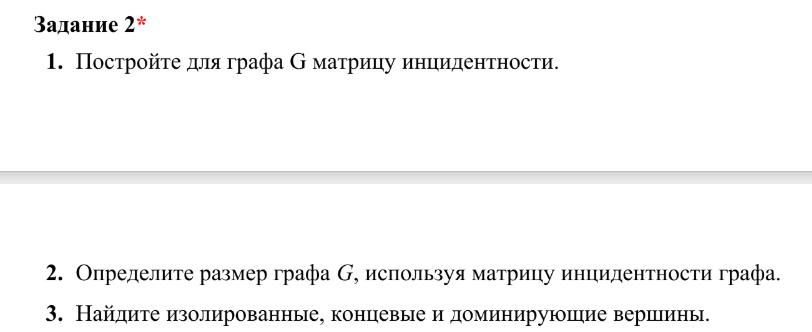
к.т.н., Деев М.В.

Пенза 2025

**Цель работы** – Освоение методов программной обработки графовых структур путем реализации алгоритмов преобразования матричных представлений графов и анализа их структурных характеристик.

**Лабораторное задание:**

****



**Код программы**

**C#**

using System;

using System.Collections.Generic;

public class proekt

{

static int CountEdges(int[,] G)

{

int count = 0;

for (int i = 0; i < G.GetLength(0); i++)

{

for (int j = i + 1; j < G.GetLength(0); j++)

{

count += G[i, j];

}

}

return count;

}

static int[,] BuildIncidenceMatrix(int[,] adjacencyMatrix)

{

int vertices = adjacencyMatrix.GetLength(0);

int edges = CountEdges(adjacencyMatrix);

int[,] incidenceMatrix = new int[vertices, edges];

int edgeIndex = 0;

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

{

for (int j = i + 1; j < vertices; j++)

{

if (adjacencyMatrix[i, j] == 1)

{

incidenceMatrix[i, edgeIndex] = 1;

incidenceMatrix[j, edgeIndex] = 1;

edgeIndex++;

}

}

}

return incidenceMatrix;

}

static void AnalyzeVertices(int[,] matrix)

{

int size = matrix.GetLength(0);

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

{

int degree = 0;

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

{

degree += matrix[i, j];

}

switch (degree)

{

case 0:

Console.WriteLine($"Вершина {i}: Изолированная");

break;

case 1:

Console.WriteLine($"Вершина {i}: Концевая");

break;

default:

if (degree == size - 1)

{

Console.WriteLine($"Вершина {i}: Доминирующая");

}

else

{

Console.WriteLine($"Вершина {i}: Обычная (степень {degree})");

}

break;

}

}

}

static void PrintMatrix(int[,] matrix, string name)

{

Console.WriteLine($"\n{name}:");

int rows = matrix.GetLength(0);

int cols = matrix.GetLength(1);

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

{

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

{

Console.Write(matrix[i, j] + " ");

}

Console.WriteLine();

}

}

public static void Main()

{

Console.Write("Введите размер матрицы: ");

int size = Convert.ToInt32(Console.ReadLine());

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

Random rnd = new Random();

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

{

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

{

if (i == j)

{

G[i, j] = 0;

}

else

{

int value = rnd.Next(0, 2);

G[i, j] = value;

G[j, i] = value;

}

}

}

PrintMatrix(G, "Матрица смежности G");

Console.WriteLine($"\nКоличество ребер: {CountEdges(G)}");

PrintMatrix(BuildIncidenceMatrix(G), "Матрица инцидентности");

int graphSize = BuildIncidenceMatrix(G).GetLength(1);

Console.WriteLine($"\nРазмер графа G (по матрице инцидентности): {graphSize} ребер");

Console.WriteLine("\nАнализ вершин:");

AnalyzeVertices(G);

}

}

**Результаты работы программы**

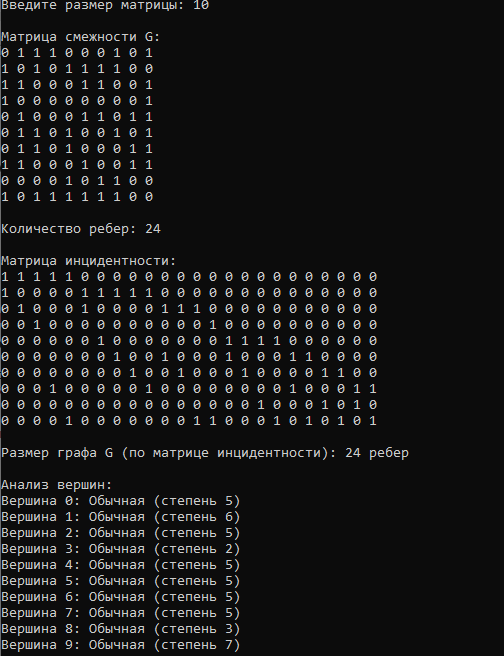


Рисунок 1 - Результат работы программы

**Вывод:** В результате выполнения работы были успешно реализованы алгоритмы преобразования матрицы смежности в матрицу инцидентности и разработаны методы анализа вершин графа.