Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

Пермский национальный исследовательский политехнический университет

Кафедра «Информационные технологии и автоматизированные системы»

**ОТЧЕТ**

Дисциплина: «Дискретная математика»

Тема: «Минимальное остовное дерево»

Семестр 4

Выполнил работу

Студент группы ИВТ-23-2б

Меновщиков Глеб Николаевич

Проверил

старший преподаватель кафедры ИТАС

Рустамханова Гульшат Ильдаровна

Пермь 2025

**Постановка задачи**

Написать программу, которая находит минимальное остовное дерево в заданном графе, используя два алгоритм Прима.

**Входные данные:**

Граф задаётся матрицей смежности размером 10×10, которая считывается из текстового файла.

**Требования к программе:**

Программа должна считывать матрицу смежности из файла (имя файла вводится пользователем).

Выводить на экран считанную матрицу смежности в удобочитаемом виде.

Обеспечить корректную обработку ошибок (например, если файл не найден).

Реализовать алгоритм Прима для нахождения MST и вывести результат:

**Выходные данные:**

Список рёбер MST и их веса.

Общий вес MST.

**Код программы**

using System;

public class Prima

{

private int FindMinKey(int[] keys, bool[] inMst, int size)

{

int minVal = int.MaxValue;

int minIdx = -1;

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

{

if (!inMst[v] && keys[v] < minVal)

{

minVal = keys[v];

minIdx = v;

}

}

return minIdx;

}

public void PrimMST(int[,] graph)

{

int size = graph.GetLength(0);

int[] keys = new int[size];

int[] parent = new int[size];

bool[] inMst = new bool[size];

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

{

keys[i] = int.MaxValue;

inMst[i] = false;

}

keys[0] = 0;

parent[0] = -1;

for (int count = 0; count < size - 1; count++)

{

int u = FindMinKey(keys, inMst, size);

if (u == -1) break;

inMst[u] = true;

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

{

if (graph[u, v] > 0 && !inMst[v] && graph[u, v] < keys[v])

{

parent[v] = u;

keys[v] = graph[u, v];

}

}

}

PrintResult(parent, graph, size);

}

private void PrintResult(int[] parent, int[,] graph, int size)

{

int total = 0;

Console.WriteLine("Ребро\tВес");

for (int i = 1; i < size; i++)

{

if (parent[i] < 0 || parent[i] >= size) continue;

int weight = graph[i, parent[i]];

Console.WriteLine($"{parent[i] + 1} - {i + 1}\t{weight}");

total += weight;

}

Console.WriteLine($"Общий вес МОД: {total}\n");

}

}

public static class GraphProgram

{

public static string[] VertexNames = { "a", "b", "c", "d", "e", "f", "g", "h", "i", "j" };

public static int[,] GetMatrixFromFile(string fileName)

{

int[,] matrix = new int[10, 10];

try

{

string[] lines = File.ReadAllLines(fileName + ".txt");

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

foreach (string line in lines)

{

string[] tokens = line.Split(new[] { ' ' }, StringSplitOptions.RemoveEmptyEntries);

foreach (string token in tokens)

{

if (int.TryParse(token, out int num))

{

numbers.Add(num);

}

}

}

int index = 0;

for (int i = 0; i < 10; i++)

{

for (int j = 0; j < 10; j++)

{

matrix[i, j] = index < numbers.Count ? numbers[index++] : 0;

}

}

}

catch

{

Console.WriteLine("Файл не читается :((((((");

}

return matrix;

}

public static void PrintMatrix(int[,] matrix)

{

Console.Write(" ");

foreach (var name in VertexNames)

{

Console.Write(name.PadLeft(3));

}

Console.WriteLine();

for (int i = 0; i < 10; i++)

{

Console.Write(VertexNames[i] + " ");

for (int j = 0; j < 10; j++)

{

Console.Write(matrix[i, j].ToString().PadLeft(3));

}

Console.WriteLine();

}

}

}

public class Program

{

public static void Main(string[] args)

{

while (true)

{

Console.WriteLine("Имя файла:");

string fileName = Console.ReadLine().Trim();

int[,] matrix = GraphProgram.GetMatrixFromFile(fileName);

Console.WriteLine("\nМатрица");

GraphProgram.PrintMatrix(matrix);

Console.WriteLine("\nАлгоритм Прима");

Prima prima = new Prima();

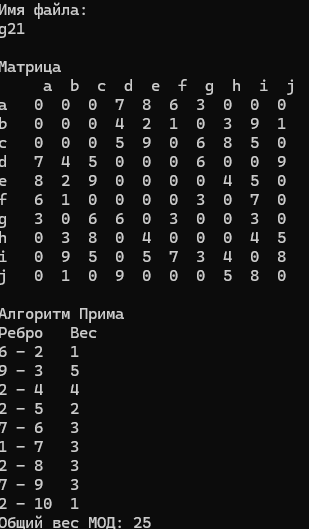
prima.PrimMST(matrix);

}

}

}

**Результаты работы программы**



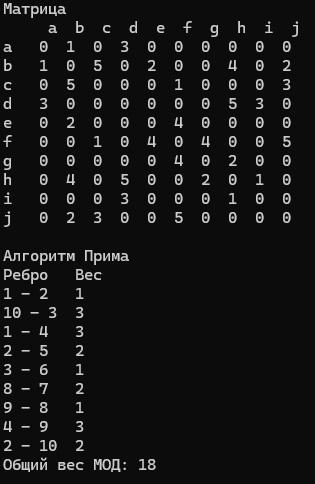
****

Рисунок 1-2 - результаты работы программы