МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования

"МИРЭА - Российский технологический университет"

РТУ МИРЭА

Институт информационных технологий (ИИТ)
Кафедра МОиСИТ

ОТЧЕТ
ПО ПРАКТИЧЕСКОЙ РАБОТЕ №7.2
«НЕЛИНЕЙНЫЕ СТРУКТУРЫ»

по дисциплине

«Структуры и алгоритмы обработки данных»

Выполнил студент группы ИКБО-10-24 Таганов А.А.

Москва 2025

Цель работы: Составить программу создания графа и реализовать процедуру для работы с графом, определенную индивидуальным вариантом задания.

Реализуемый алгоритм: Построение остовного дерева алгоритмом Прима.

Описание решения: Алгоритм Прима работает следующим образом. Берется произвольная начальная вершина. Считается, что у нас есть множество вершин, уже включенных в дерево (множество T). На каждом шаге осматриваются все ребра, идущие из T к вершинам вне него, среди них выбирается ребро минимального веса, это ребро добавляется в остов, добавляется вершина в множество. Цикл повторяется, пока не будут включены все вершины. Если в какой-то момент нет ребер, ведущих наружу, то граф не связен.

Граф создается в программе с помощью функции createGraphAdjMatrix() путем ввода матрицы смежности графа. Функция возвращает вектор векторов типа int.

Листинг 1 – Функция создания графа

```
vector<vector<int>> createGraphAdjMatrix() {
    int n;
    cout << "Enter number of graph's verticles: ";
    cin >> n;

    vector<vector<int>> g(n, vector<int>(n));
    cout << "Enter adjacency matrix" << n << " lines with " << n
         << " integers, 0 means absence of an edge:\n";

    for (int i = 0; i < n; ++i) {
        for (int j = 0; j < n; ++j) {
            cin >> g[i][j];
        }
    }

    return g; // возвращаем созданный граф
}
```

Алгоритм принимает матрицу смежности, введенную пользователем. Задается вектор для хранения минимального веса, где все значения задаются большим числом. Задается вектор для хранения ребер типа tuple. Реализуется алгоритм Прима и возвращается остовное дерево (вектор ребер) и суммарный вес дерева.

Листинг 2 – Алгоритм Прима

```
void primMST(const vector<vector<int>>& g) {
    const int INF = 1000000000;
    int n = g.size();

    vector<int> minEdge(n, INF);     // минимальный вес ребра, ведущего в вершину
    vector<int> selEdge(n, -1);     // откуда пришли в вершину
    vector<bool> used(n, false);   // включена ли вершина в остов

    minEdge[0] = 0; // начинаем с вершины 0 (можно выбрать любую)

    vector<tuple<int,int,int>> mstEdges; // (u, v, w)
    int totalWeight = 0;

    for (int i = 0; i < n; ++i) {
        int v = -1;
        //  выбираем неиспользованную вершину с минимальным minEdge
        for (int j = 0; j < n; ++j) {
            if (!used[j] && (v == -1 || minEdge[j] < minEdge[v])) {
                v = j;
            }
        }

        if (minEdge[v] == INF) {
            cout << "Graph is disconnected, it's not possible to build a spanning tree.\n";
            return;
        }

        used[v] = true;

        // добавляем ребро в остов (кроме стартовой вершины)
        if (selEdge[v] != -1) {
            int u = selEdge[v];
            int w = g[u][v];
            mstEdges.push_back({u, v, w});
            totalWeight += w;
        }
```

```
        // обновляем расстояния до остальных вершин
        for (int to = 0; to < n; ++to) {
            int w = g[v][to];
            if (w > 0 && !used[to] && w < minEdge[to]) {
                minEdge[to] = w;
                selEdge[to] = v;
            }
        }
    }
```

Программа была протестирована на графе, взятом из задания согласно варианту. Граф представлен на рисунке 1.

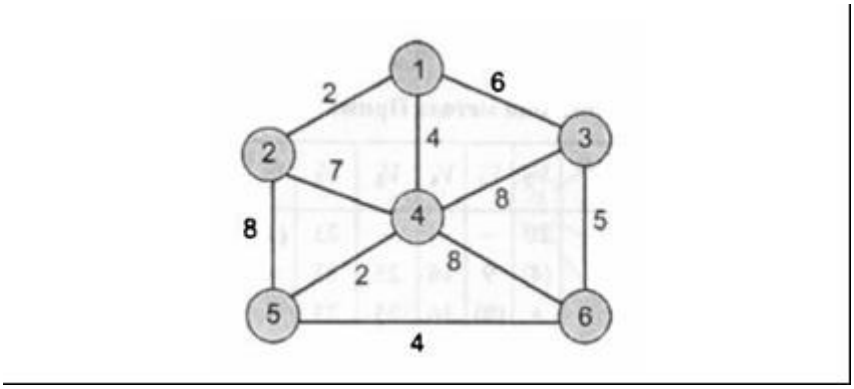

Рисунок 1 – Граф, представленный в задании

Для тестирования была составлена матрица смежности, далее поданная на вход разработанной программе

```
Enter number of graph's verticles: 6
Enter adjacency matrix6 lines with 6 integers, 0 means absence of an edge:
0 2 6 4 0 0
2 0 0 7 8 0
6 0 0 8 0 5
4 7 8 0 2 8
0 8 0 2 0 4
0 0 5 8 4 0
```

Рисунок 2 – Пользовательский ввод

Вывод программы представлен на рисунке 3.

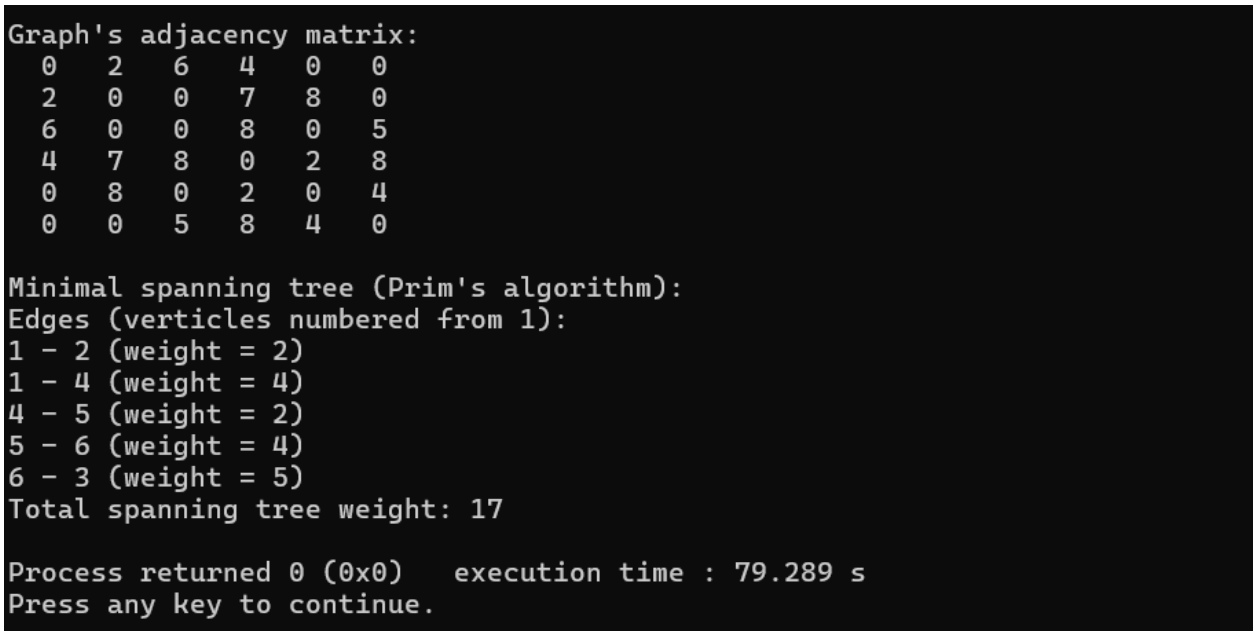

Рисунок 3 – Вывод программы

Вывод

Поставленные задачи выполнены: изучены и реализованы алгоритмы по работе с взвешенным неориентированным графом. Реализован алгоритм построения остовного дерева методом Прима.

Литература.

1. Страуструп Б. Программирование. Принципы и практика с использованием C++. 2-е изд., 2016.

2. Документация по языку C++ [Электронный ресурс]. URL: https://docs.microsoft.com/ruru/cpp/cpp/ (дата обращения 13.11.2025).

3. Курс: Структуры и алгоритмы обработки данных. Часть 2 [Электронный ресурс].URL:https://online-edu.mirea.ru/course/view.php?id=4020 (дата обращения 13.11.2025).