МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА № 25

КУРСОВАЯ РАБОТА (ПРОЕКТ) ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

кандидат технических наук Е.М. Линский

должность, уч. степень, звание подпись, дата инициалы, фамилия

КУРСОВАЯ РАБОТА (ПРОЕКТ) Алгоритм Прима

по дисциплине: ОСНОВЫ ПРОГРАММИРОВАНИЯ

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР. № 2154 Д.А. Раупов

подпись, дата инициалы, фамилия

Санкт-Петербург 2022

Оглавление

[1 Введение 3](#_Toc120617365)

[2 Постановка задачи 4](#_Toc120617366)

[3 Алгоритм 5](#_Toc120617369)

[3.1 Описание алгоритма 5](#_Toc120617370)

[3.2 Описание работы алгоритма 6](#_Toc120617371)

[3.3 Описание структуры 6](#_Toc120617372)

[3.4 Псевдокод и анализ сложности 7](#_Toc120617373)

[4 Инструкция пользователя 9](#_Toc120617374)

[5 Тестовые примеры 10](#_Toc120617375)

[Список используемых источников 12](#_Toc120617376)

# Введение

Задачей данной курсовой работы является реализация алгоритма Прима на языке c++, с использованием утилиты графической обработки графов graphviz. Алгоритм Прима – алгоритм построения минимального остовного дерева взвешенного связанного неориентированного графа. Алгоритм впервые был открыт в 1930 году чешским математиком Войцехом Ярником, позже переоткрыт Робертом Примом в 1957 году[[1](#_Список_используемых_источников)]

# Постановка задачи

В ходе курсового проекта требуется решить следующие задачи:

* Реализовать алгоритм Прима
* Реализовать ввод и вывод данных программы через файлы
* Реализовать вывод графов в файлы для работы с graphviz
* Реализовать работы программы через командную строку

# Алгоритм

* 1. Описание алгоритма

На вход алгоритма подается взвешенный неориентированный граф. Сначала берется произвольная вершина и находится инцидентное ей ребро с наименьшей стоимостью. Ребро с соединёнными в нем двумя вершинами образует дерево, затем рассматриваются ребра графа, одна вершина которого уже принадлежит дереву, а другая нет, из этих ребер выбираем ребро наименьшей стоимости. Выбранное ребро на каждом шаге присоединяется к дереву. Рост дерева продолжается до тех пор, пока не будут исчерпаны все вершины исходного графа.

Результатом работы алгоритма является минимальное остовное дерево.

* 1. Описание работы алгоритма

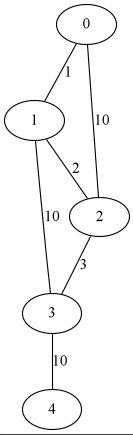


Рисунок 1- исходный граф

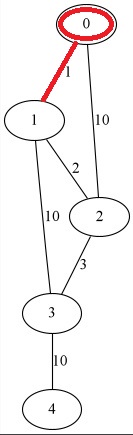


Рисунок 2 - 1 шаг алгоритма

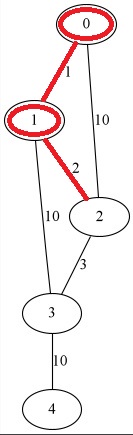


Рисунок 3- 2 шаг алгоритма

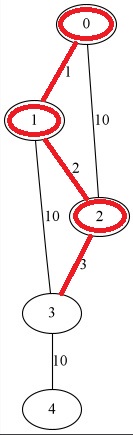


Рисунок 4- 3 шаг алгоритма

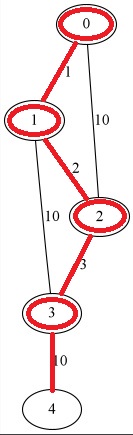


Рисунок 5 - 4 шаг алгоритма

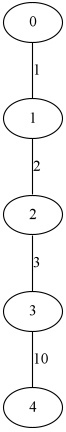


Рисунок 6- минимальное остовное дерево

Первым шагом алгоритм берет вершину 0 как начало минимального остовного дерева, после ищет ребро наименьшей стоимости и соединяет две вершины, образуя дерево, алгоритм продолжается пока не закончатся все вершины графа.

* 1. Описание структуры

|  |  |
| --- | --- |
| **public:** | |
| void allocate\_mem(int n); | Выделение памяти |
| MST(const string& inFile); | Конструктор класса |
| ~MST(); | Деструктор класса |
| void prim(int startVert); | Функция алгоритма Прима |
| int getVerts() const; | Получить количество вершин |
| int getEdges() const; | Получить количество ребер |
| void printGraph(const string &outFile); | Функция вывода графа в файл |
| void draw\_graph(const string& FileName1, const string& FileName2); | Функция создания dot файлов для графа |

* 1. Псевдокод и анализ сложности

Ниже приведен псевдокод алгоритма

Prim (число startVert)

{

Число x,y,min;  
 пока(число i < количество вершин)  
 пока(число j < количество вершин)

{

Selected[i] = false;

Graph2[i][j] = 0;

}

}

Selected[startVert] = true;

Пока(число е < количество ребер)

{

x = 0; y = 0; min = ∞;

пока(число i < количество вершин)

{

Пока(число j < количество вершин и selected[i] = true)

{

Если( selected[j] = false и graph[i][j] > 0 и min > graph[i][j])

то{

min = graph[i][j];

x = I; y = j;

}

}

}

graph2[x][y] = graph[x][y];  
 graph2[y][x] = graph[x][y];  
 selected[y] = true;

}

};

На основе этого псевдокода можно проанализировать сложность

написанного алгоритма. Пусть n – количество ребер, m – количество вершин, тогда сложность написанного алгоритма будет равняться O(*m2+n*).

# Инструкция пользователя

Для начала пользования программой пользователю необходимо создать файл с матрицей смежности.

Пример:

0 1 10 0 0

1 0 2 10 0

10 2 0 3 0

0 10 3 0 10

0 0 0 10 0

После пользователю необходимо открыть командную строку и прописать.

cd репозиторий с программой/prim.exe

prim.exe ВходнойФайл.txt ВыходнойФайл.txt ВыходнойФайлИзначальногоГрафа.dot ВыходнойФайлМинимальногоОстовногоДерева.dot

После пользователю открывается меню программы, с возможностью выбора функций, которыми он хочет воспользоваться.

What do you want to do?

1. Use Prim's algorithm

2. Output the number of edges

3. Output the number of verts

4. Print the graph to a file

5. Draw a graph to a file

6. Exit

1 – Использование алгоритма Прима

2 – Вывести количество ребер

3 – Вывести количество вершин

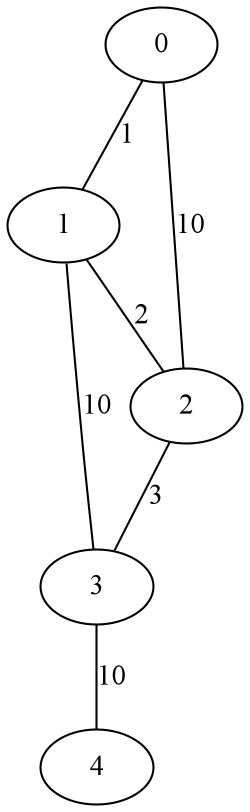
4 – Напечатать граф в файл

5 – Нарисовать граф в файл

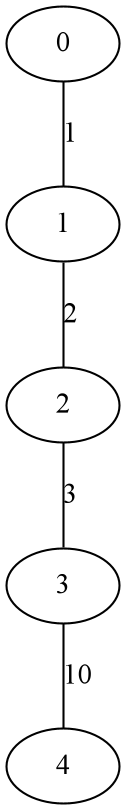
После использований функций 4-5 пользователь получит на выход файлы, с результатом работы программы.

# Тестовые примеры

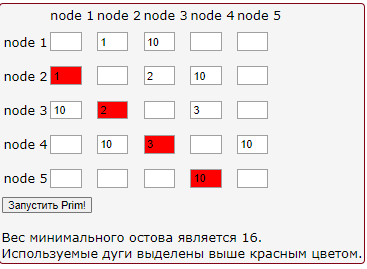
Я с помощью своей программы рассчитал минимальное остовное дерево у 5 графов и проверил их с помощью онлайн решения алгоритма Прима[[3](#_Список_используемых_источников)].

1. 

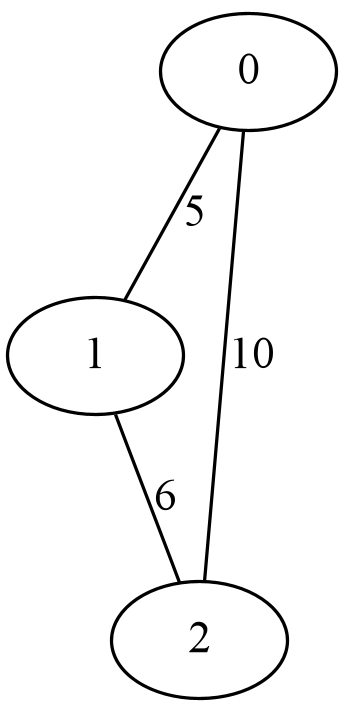
Исходный граф



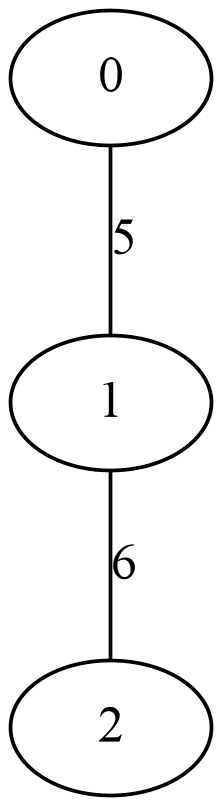
*Минимальное остовное дерево*

**

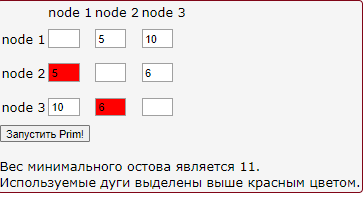
Проверка

1. 

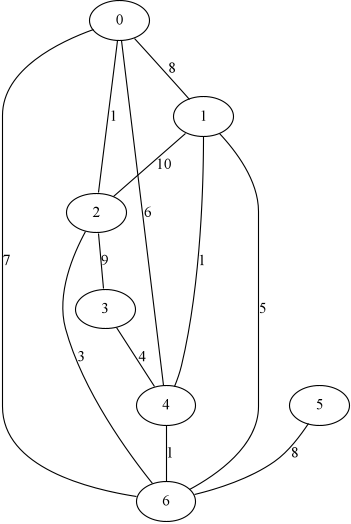
Исходный граф



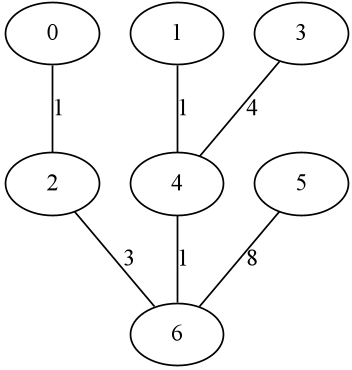
Минимальное остовное дерево



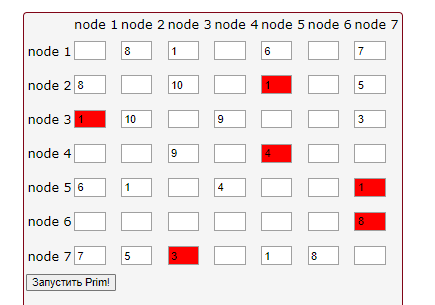
Проверка

1. 

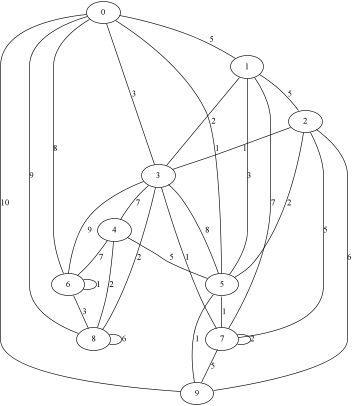
Исходный граф



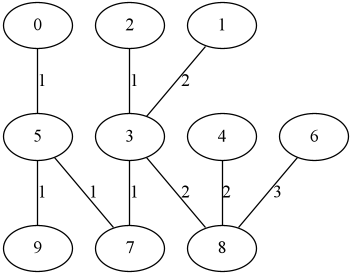
Минимальное остовное дерево



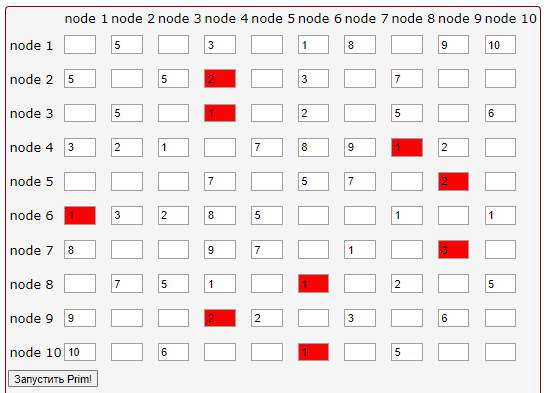
Проверка

1. 

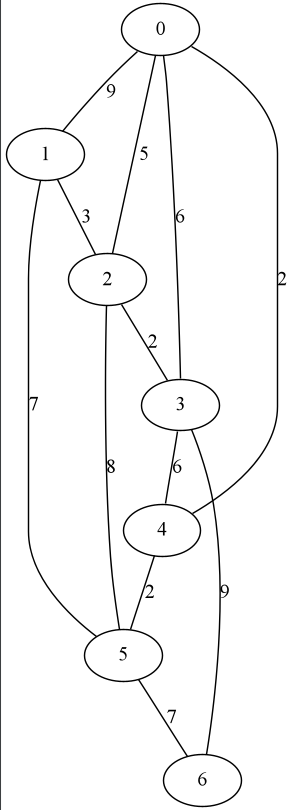
Исходный граф



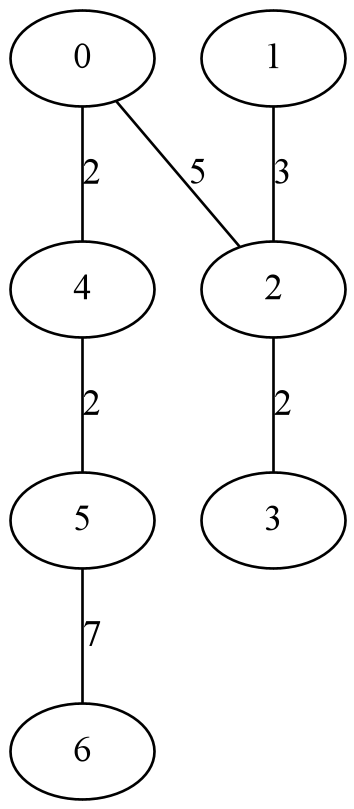
Минимальное остовное дерево



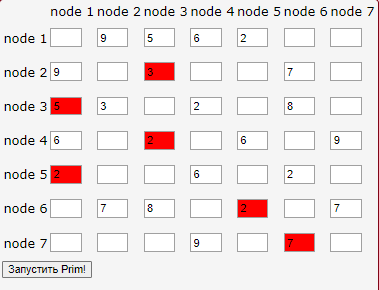
Проверка

1. 

Исходный граф

**

Минимальное остовное дерево



Проверка

# Список используемых источников

1. Алгоритм Прима // neerc.ifmo URL: <https://neerc.ifmo.ru/> (дата обращения: 2022).
2. Теория графов // wikipedia URL: <https://ru.wikipedia.org/>
3. Онлайн алгоритм прима//intellect.icu URL:<https://intellect.icu/> (дата обращения: 2022).