# Московский авиационный институт
# (национальный исследовательский университет)

## Факультет информационных технологий и прикладной математики

## Кафедра вычислительной математики и программирования


### Лабораторная работа №9 по курсу «Дискретный анализ»


| | |
|---|---|
| Студент: | И. П. Попов |
| Преподаватель: | А. А. Кухтичев |
| Группа: | М8О-306Б |
| Дата: | |
| Оценка: | |
| Подпись: | |



**Москва, 2022**

# Лабораторная работа №9

**Задача:** Задан взвешенный неориентированный граф, состоящий из n вершин и m ребер. Вершины пронумерованы целыми числами от 1 до n. Необходимо найти длину кратчайшего пути из вершины с номером start в вершину с номером finish при помощи алгоритма Дейкстры. Длина пути равна сумме весов ребер на этом пути. Граф не содержит петель и кратных ребер.

### Формат ввода

В первой строке заданы $1 \leqslant n \leqslant 10^5 1 \leqslant m \leqslant 10^5, 1 \leqslant start \leqslant n1 \leqslant finish \leqslant n.m.\breve{}, , .\breve{} 010^9$.

### Формат вывода

Необходимо вывести одно число – длину кратчайшего пути между указанными вершинами. Если пути между указанными вершинами не существует, следует вывести строку "No solution"(без кавычек).

# 1 Описание

**Алгоритм Дейкстры** (англ. Dijkstra's algorithm) — алгоритм на графах, изобретённый нидерландским учёным Эдсгером Дейкстрой в 1959 году. Находит кратчайшие пути от одной из вершин графа до всех остальных. Алгоритм работает только для графов без рёбер отрицательного веса. Алгоритм широко применяется в программировании, например, его используют протоколы маршрутизации OSPF и IS-IS.

Каждой вершине из V сопоставим метку — минимальное известное расстояние от этой вершины до a. Алгоритм работает пошагово — на каждом шаге он «посещает» одну вершину и пытается уменьшать метки. Работа алгоритма завершается, когда все вершины посещены.

**Инициализация**
Метка самой вершины a полагается равной 0, метки остальных вершин — бесконечности. Это отражает то, что расстояния от a до других вершин пока неизвестны. Все вершины графа помечаются как непосещённые.

**Шаг алгоритма**
Если все вершины посещены, алгоритм завершается. В противном случае, из ещё не посещённых вершин выбирается вершина u, имеющая минимальную метку. Мы рассматриваем всевозможные маршруты, в которых u является предпоследним пунктом. Вершины, в которые ведут рёбра из u, назовём соседями этой вершины. Для каждого соседа вершины u, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки u и длины ребра, соединяющего u с этим соседом.

Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину u как посещённую и повторим шаг алгоритма.

# 2 Исходный код

Фрагмент кода, где реализован алгоритм Дейкстры, с использованием очереди с приорететом:

```
while(!prior.empty()){
    int minDistVert = prior.top().to;
    prior.pop();

    if(used[minDistVert]){
        continue;
    }

    if(minDistVert == LINF){
        break;
    }

    used[minDistVert] = true;

    for (auto [to, cost]: graph[minDistVert]){
        ll newDist = cost + dist[minDistVert];

        if(dist[to] > newDist){
            dist[to] = newDist;
            prior.push(Edge(to, newDist));
        }
    }
}
```

# 3 Консоль

tmp:

```
5 6 1 5
1 2 2
1 3 0
3 2 10
4 2 1
3 4 4
4 5 5
```

console output:

```
8
```

# 4 Тест производительности

Засечем время выполнения программы для случайного графа.
В результате работы benchmark.cpp видны следующий результат:

```
root@Lunidep:"/Desktop/DA/lab9$ ./bench <test_graph
Time: 0.002339
```

# 5 Выводы

Выполнив девятую лабораторную работу по курсу «Дискретный анализ», я больше узнал о графах и алгоритмах работы с ними. Теория графов к настоящему времени содержит достаточно много эффективных инструментов для решения столь же широкого круга проблем. Однако, средства и идеи, сегодня относящиеся к области дискретной математики, именуемой теорией графов, пребывают по сей момент в некотором единстве.

Так, в решаемой мной задаче , можно заметить, что граф неориентированный, и можно использовать обход в глубину. При запуске обхода из вершины, принадлежащей к некоторой компоненте связности, обход посетит все вершины из этой компоненты и только их. Таким образом, в функцию обхода можно передавать вектор, в который будут помещаться вершины из очередной компоненты связности.

Сложность совпадает со сложностью обхода в глубину, то есть O(V + E).

Существуют и другие алгоритмы поиска кратчайшего пути от одной вершины графа до другой. Например, алгоритм Форда-Беллмана. Он имеет сложность O(V * E), но умеет работать с ребрами, имеющими отрицательный вес. Алгоритм Дейкстры же является жадным и имеет сложность O(VlogV).

# Список литературы


[1] Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. *Алгоритмы: построение и анализ, 2-е издание.* — Издательский дом «Вильямс», 2007. Перевод с английского: И. В. Красиков, Н. А. Орехова, В. Н. Романов. — 1296 с. (ISBN 5-8459-0857-4 (рус.))