**Содержание**

[Задание 2](#_Toc57920040)

[Математическая формулировка задачи 3](#_Toc57920041)

[Текст программы на языке С++ 4](#_Toc57920043)

[Список литературы 5](#_Toc57920048)

# **Задание**

Имеется некоторое количество авиарейсов между городами мира, для каждого известна стоимость. Стоимость перелёта из A в B может быть не равна стоимости перелёта из B в A. Найти маршруты минимальной стоимости между городами (возможно, с пересадками).

**Формат входных данных**

Во входном файле записано сначала число N (1<=N<=100), определявшее количество рассматриваемых городов. Затем идет число M - количество авиарейсов между городами мира, далее идет описание самих дорог. Каждое сообщение между городами задается тремя числами - номерами городов и стоимостью перелета.

**Формат выходных данных**

На экран выведите числа - суммарные стоимости маршрутов с указанием номеров городов или -1, если добраться невозможно.

# **Математическая формулировка задачи**

***Алгоритм Форда - Беллмана.***

Основными вычисляемыми величинами этого алгоритма являются величины **λi(к)**, где i = 1,2, ... , n (n - число вершин графа); к = 1, 2, ... , n - 1. Для фиксированных **i** и **к** величина λi(k) равна длине минимального пути, ведущего из заданной начальной вершины **х1** в верши­ну **xi** и содержащего не более **к** дуг.

**Шаг 1**. Установка начальных условий.

Определяем число вершин графа **n** и матрицу весов С= (сij).

**Шаг 2**. Положим к = 0. Положим λi(О) = ∞ для всех вершин, кроме хi, положим λi(0)= 0.

**Шаг 3.** В цикле по k, k =1,.., n - 1, каждой вершине хi на k -ом шаге приписать индекс λi(k) по следующему правилу:

https://studfiles.net/html/2706/399/html_OUd5L6Sx4g.By1g/img-B0shns.png  **(1)**

для всех вершин, кроме x1, положить λ1(k) = О

В результате работы алгоритма формируется таблица индексов λ1(k), i=1, 2, ... , n, k = 0, 1, 2, ... , n - 1. При этом λ1(k) определяет длину минимального пути из первой вершины в i-ую, содержащего не более k дуг.

**Шаг 4.** Восстановление минимального пути.

Для любой вершины xs предшествующая ей вершина хr определяется из соотношения:

λr(n-2)+crs= λs(n-1), xrhttps://studfiles.net/html/2706/399/html_OUd5L6Sx4g.By1g/img-P3OSS5.pngG-1(xs), **(2)**

где G-1(xs) - прообраз вершины xs.

Для найденной вершины хr предшествующая ей вершина хq определяется из соотно­шения:

λq(n-3)+cqr= λr(n-2), xqhttps://studfiles.net/html/2706/399/html_OUd5L6Sx4g.By1g/img-RYAteo.pngG-1(xr),

где G-1(xr) - прообраз вершины хr и т. д.

Последовательно применяя это соотношение, начиная от последней вершины хi, найдем минимальный путь.

# **Текст программы на языке Python**

N = int(input())

M = int(input())

a = [[float('inf')]\*N for i in range(N)]

for i in range(N):

a[i][i] = 0

for i in range(M):

b = input().split()

a[int(b[0])-1][int(b[1])-1] = int(b[2])

for i in range(N):

for ii in range(N):

for iii in range(N):

a[ii][iii] = min(a[ii][iii], a[ii][i] + a[i][iii])

for i in range(N):

for j in range(N):

if a[i][j] != float('inf'):

print(f"из {i} в {j}:{a[i][j]}")

else:

print(f"из {i} в {j}:-1")

# **Список литературы**

1. Викиконспекты – Алгоритм Флойда([https://neerc.ifmo.ru/wiki/index.php) 15.05.2023](https://neerc.ifmo.ru/wiki/index.php)%2015.05.2023)
2. «Изучаем Python», Марк Лутц