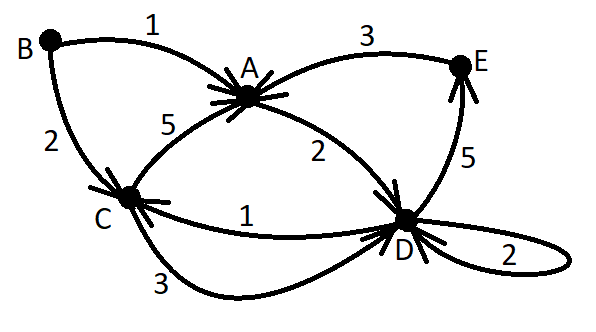
Домашнее задание №1

## C:\Users\karpeta\Desktop\Граф1.png

1.1. Найдите кратчайшие расстояния и и кратчайшие пути от вершины V5 до всех остальных вершин в графе, приведенном выше, с использованием алгоритма Дейкстры.

1.2. Найдите кратчайшие пути между всеми парами вершин в графе, приведенном выше, с использованием алгоритма Флойда.



2.1. Найдите кратчайшие расстояния и и кратчайшие пути от вершины A до всех остальных вершин в графе, приведенном выше, с использованием алгоритма Дейкстры.

2.2. Найдите кратчайшие пути между всеми парами вершин в графе, приведенном выше, с использованием алгоритма Флойда.

3. Самостоятельно изучите метод восстановления кратчайших путей при использовании алгоритма Флойда, используя соответствующие статьи на [neerc.ifmo.ru](https://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0#.D0.92.D1.8B.D0.B2.D0.BE.D0.B4_.D0.BA.D1.80.D0.B0.D1.82.D1.87.D0.B0.D0.B9.D1.88.D0.B5.D0.B3.D0.BE_.D0.BF.D1.83.D1.82.D0.B8) или в [википедии](https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0_%E2%80%94_%D0%A3%D0%BE%D1%80%D1%88%D0%B5%D0%BB%D0%BB%D0%B0" \l "%D0%A0%D0%B5%D0%BA%D0%BE%D0%BD%D1%81%D1%82%D1%80%D1%83%D0%BA%D1%86%D0%B8%D1%8F_%D0%BF%D1%83%D1%82%D0%B5%D0%B9).

Домашнее задание №2

**"Представление графов в памяти ЭВМ"**

- реализовать представление ВЗВЕШЕННОГО ОРИЕНТИРОВАННОГО графа 3мя способами

- 6 преобразований между всеми парами форматов

- реализовать ввод и вывод каждого из представлений из/в файл в следующих форматах:

матрица смежности (вес соответствующего ребра или 0):

4 (количество вершин)

0 10 20 30

0  0 40  0

0 50  0  0

10 0 10  0

список ребер (откуда, куда, вес):

4 7 (количество вершин, количество ребер)

1 2 10

1 3 20

1 4 30

2 3 40

3 2 50

4 1 10

4 3 10

список смежности (для каждой вершины - количество смежных, затем пары индекс-вес)

4 (количество вершин)

3   2 10   3 20   4 30 (первая вершина смежна с 3мя другими, затем 3 пары - индекс соотв. смежной и вес соотв. ребра)

1   3 40

1   2 50

2   1 10   3 10

Домашнее задание №3

1)  **Заячья избушка**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Заяц решил построить себе новую избушку на одной из полян в лесу. Всего в лесу N

полян, связанных между собой K тропинками разной длины. Между двумя полянами проложено не более одной тропинки. Тропинки не пересекаются между собой, так как прокладывались с использованием переходных мостов и подземных переходов. На полянах A и B уже построили себе домики Волк и Лиса, а Зайцу, естественно, хочется жить как можно дальше от них. Требуется выбрать полянку для строительства избушки, чтобы величина 1a+1b была минимальной, где a – наименьшее расстояние по тропинкам от домика Зайца до домика Волка, а b

– наименьшее расстояние от домика Зайца до домика Лисы. Если нет пути между полянами, то соответствующее слагаемое принимаем равным 0.

*Ввод*

Во входном файле в первой строке четыре целых числа через один пробел: N, K, A

и B – количество полян и тропинок, номера полян с домиками Волка и Лисы (3≤N≤20, 1≤K≤100, 1≤A≤N, 1≤B≤N). Далее следует K

строк с информацией о тропинках. В каждой строке содержится три целых числа через пробел: номера двух полянок, связанных тропинкой, и длина тропинки (от 1 до 1000).

*Вывод*

В выходной файл вывести все полянки с минимальным значением указанного выражения в порядке возрастания номеров. Каждый номер поляны выводить на отдельной строке.

*Пример ввода:*

5 4 2 3

2 4 7

4 5 1

1 5 4

1 3 8

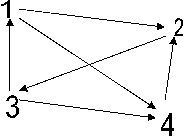
*Пример вывода*

1

5

**2) Столица**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

В тридевятом царстве, в тридевятом государстве между двумя любыми городами есть дорога, но только в одну сторону. Местонахождение резиденции царя держится в секрете, но известно, что в столичный город можно попасть из любого другого города, проезжая не более чем через один промежуточный город.

Напишите программу, которая определит местонахождение столицы по информации о дорогах страны.

В первой строке входного файла содержится одно целое число N

(1 ≤ N ≤ 100) – количество городов. Далее следует (N−1) строка, содержащих только символы + (плюс) и – (минус). Длина i-й строки равна i−1. j-й символ в i-й строке показывает направление дороги: + означает, что дорога ведет из города j в город i;  – означает, что дорога ведет из города i в город j

.

В выходной файл в первой строке вывести количество городов-кандидатов для местонахождения столицы, а во второй строке – номера городов-кандидатов в порядке возрастания, разделяя их пробелами.

*Пример ввода*

4

+

-+

+-+

*Пример вывода*

3

2 3 4

**3) Fiat lux!**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Сэм решил украсить дом к Рождеству иллюминацией. Он соединил проводами несколько ламп и подключил их к электросети. Затем Сэм повернул рубильник, и весь район погрузился в темноту.

Напишите программу, которая определит, используя информацию о выполненных соединениях, произойдет ли короткое замыкание при включении рубильника. Сопротивление соединительных проводов считать нулевым.

В первой строке входного файла содержатся два целых числа, разделенных пробелами – количество ламп N

(1 ≤ N ≤ 100) и количество проводов K (1 ≤ K ≤ 1000). Далее следует K строк, в каждой строке сначала указывается номер элемента (0 для электросети, от 1 до N

для лампы), к которому был присоединен один из концов провода, потом через пробел – номер контакта 1 или 2 (все элементы имеют два контакта), затем указывается элемент и контакт, к которому был присоединен другой конец провода, в том же формате.

В выходной файл в первой строке вывести сообщение "YES", если произойдет короткое замыкание (ситуация, в которой существует цепь от одного контакта электросети до другого, не содержащая нагрузки – ламп), или сообщение "NO", если короткого замыкания не будет.

Пример ввода:

3 4

0 1 1 1

1 2 2 1

2 2 0 2

1 2 0 2

Пример вывода

NO

Домашнее задание №4

**Союз стран**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Три страны решили объединиться в союзное государство. Так как чиновникам предстоит часто ездить из одной столицы в другую, необходимо заасфальтировать некоторые дороги (в настоящий момент все дороги являются гравийными) таким образом, чтобы можно было проехать между двумя любыми столицами по хорошей дороге (необязательно, чтобы путь был кратчайшим).

Напишите программу, которая определит, какие дороги нужно заасфальтировать, чтобы общая стоимость строительства была минимальной. Стоимость строительства линейно зависит от длины асфальтируемой дороги.

*Формат ввода*

Первая строка ввода содержит два целых числа: количество городов N

(3 ≤ N ≤ 2000) и количество дорог M (N−1 ≤ M ≤ N⋅N−12). Далее следует M строк, содержащих по три целых числа: номера городов ai и bi (1 ≤ ai, bi ≤ N, ai ≠ bi), связанных гравийной дорогой, и длина дороги di (1 ≤ di ≤ 10 000

). Движение по дорогам возможно в обе стороны. Гарантируется, что существует путь между двумя любыми городами и в графе нет кратных ребер и петель. Столицами стран являются города с номерами 1, 2 и 3.

*Формат вывода*

В первой строке вывести количество асфальтируемых дорог, во второй строке — их номера (от 1

до M в соответствии со списком дорог во входном файле) через пробел в произвольном порядке. Если существует несколько вариантов, минимизирующих стоимость строительства, можно вывести любой из них.

Пример ввода:

6 7

1 6 1

1 3 10

1 5 3

2 5 6

3 4 2

3 5 4

5 6 5

Пример вывода

3

3 4 6

Домашнее задание №5

**Сеть**

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

Андрей работает системным администратором и планирует создание новой сети в своей компании. Всего будет N

хабов, они будут соединены друг с другом с помощью кабелей.

Поскольку каждый сотрудник компании должен иметь доступ ко всей сети, каждый хаб должен быть достижим от любого другого хаба – возможно, через несколько промежуточных хабов. Поскольку имеются кабели различных типов и короткие кабели дешевле, требуется сделать такой план сети (соединения хабов), чтобы максимальная длина одного кабеля была как можно меньшей. Есть еще одна проблема – не каждую пару хабов можно непосредственно соединять по причине проблем совместимости и геометрических ограничений здания. Андрей снабдит вас всей необходимой информацией о возможных соединениях хабов.

Вы должны помочь Андрею найти способ соединения хабов, который удовлетворит всем указанным выше условиям.

Первая строка входного файла содержит два целых числа: N

 – количество хабов в сети (2 ≤ N ≤ 1000) и M – количество возможных соединений хабов (1 ≤ М ≤ 15 000). Все хабы пронумерованы от 1 до N. Следующие M строк содержат информацию о возможных соединениях – номера двух хабов, которые могут быть соединены, и длина кабеля, который требуется, чтобы соединить их. Эта длина – натуральное число, не превышающее 106

. Существует не более одного способа соединить каждую пару хабов. Хаб не может быть присоединен сам к себе. Всегда существует хотя бы один способ соединить все хабы.

Сначала выведите максимальную длину одного кабеля в вашем плане соединений (это величина, которую вы должны минимизировать). Затем выведите свой план: сначала выведите P

 – количество кабелей, которые вы использовали, затем выведите P пар целых чисел – номера хабов, непосредственно соединенных в вашем плане кабелями.

Пример ввода

4 6

1 2 1

1 3 1

1 4 2

2 3 1

3 4 1

2 4 1

Пример вывода

1

3

1 2

1 3

3 4