Задача 1. BFS за линию

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан ориентированный граф G без весов. В графе есть N вершин, пронумерованных числами от 1 до N, и M дуг между некоторыми парами вершин.

Требуется найти кратчайший путь от вершины номер 1 до всех остальных вершин.

Формат входных данных

В первой строке входного файла находятся целые числа N и M — количество вершин и дуг в графе соответственно ($1 \leqslant M$, $N \leqslant 200\,000$).

Следующие M строк содержат по два числа v и u, означающие, что в графе есть дуга из вершины v в вершину $u (1 \leqslant u, v \leqslant N)$.

Формат выходных данных

В выходной файл требуется вывести ровно N строк. На i-й строке должно быть целое число:

- "-1", если не существует пути из вершины 1 до вершины номер i.
- Длина этого пути (количество рёбер) на кратчейшем пути от вершины 1 до вершины номер i.

Пример

input.txt	output.txt
5 5 1 2 2 3 3 1 2 1 3 4	0 1 2 3 -1

Задача 2. Лабиринт

Источник:	базовая
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Для заданного лабиринта найти кратчайший путь от входа до выхода.

Формат входных данных

В первой строке входного файла находятся целые числа M и N — высота и ширина лабиринта ($1 \leqslant M, N \leqslant 100$).

Каждая из следующих M строк содержит N символов, при этом символ '.' обозначает пустую клетку, символ 'X' — блок, символ 'S' — начальную клетку, символ 'F' — конечную клетку.

Формат выходных данных

Выведите в выходной файл минимальное число шагов, за которое можно добраться от начальной клетки до конечной, каждый раз переходя на соседнюю по стороне клетку и не ступая на блоки, либо число -1, если это невозможно.

Пример

input.txt	output.txt
6 7 .....X. .X..XF. ..XXXX. .X..... .X..X.. S...X..	11

Задача 3. Highways

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Островная страна Флатопия абсолютно плоская. К сожалению, во Флатопии очень бедная система общественных дорог. Правительство Флатопии знает об этой проблеме и уже построило какое-то число хайвэев, соединяющих некоторые наиболее важные города. Однако еще остались города, в которые невозможно попасть посредством хайвэев. Необходимо построить больше дорог, чтобы можно было проехать из одного города в другой, не покидая систему хайвэев.

Города Флатопии пронумерованы от 1 до N, и расположение города i задается декартовы-ми координатами (x_i, y_i). Каждый хайвэй соединяет точно два города. Все хайвэи (и те, которые есть, и те, которые нужно построить) придерживаются прямых линий, и поэтому их длина равна декартовому расстоянию между городами. Хайвэи можно использовать в обоих направлениях. Хайвэи свободно перекрывают друг друга, но водитель может перебраться с одного хайвэя на другой только в городе, который расположен на концах обоих хайвэев.

Правительство Флатопии хочет минимизировать стоимость строительства новых хайвэев. Однако они хотят, чтобы любой город мог быть достижим посредством хайвэев из любого другого. Так как Флатопия плоская, то стоимость хайвэя пропорциональна его длине. Поэтому наименее дорогой системой хайвэев будет та, которая имеет минимальную общую длину хайвэев.

Формат входных данных

Входной файл состоит из двух частей. Первая часть описывает города в стране, а вторая часть описывает все хайвэи, которые уже построены.

Первая строка входного файла содержит единственное целое число N, обозначающее количество городов($1 \leqslant N \leqslant 750$).

Каждая из следующих N строк содержит по два целых числа, x_i и y_i, разделенных пробелом. Эти значения задают координаты i-го города (для i от 1 до N). Координаты имеют абсолютные значения, не превосходящие 10000. Все города расположены в разных местах.

Следующая строка содержит одно целое число M, количество существующих хайвэев ($0 \leqslant M \leqslant 1000$).

На каждой из следующих M строк расположено по два целых числа, разделенных пробелом. Эти два числа задают номера городов, уже соединенных хайвэем. Каждая пара городов может быть соединена не более чем одним хайвэем.

Формат выходных данных

Каждая строка выходного файла должна содержать описание каждого нового хайвэя, который нужно построить, для того чтобы соединить все города с минимальной общей длиной новых хайвэев. Каждый хайвэй должен быть описан двумя номерами городов, которые он соединяет, записанными через пробел. Если ни одного нового хайвэя не нужно строить (все города уже соединены), то выходной файл должен создаваться, но быть пустым.

Пример

input.txt	output.txt
9 1 5 0 0 3 2 4 5 5 1 0 4 5 2 1 2 5 3 3 1 3 9 7 1 2	1 6 3 7 4 9 5 7 8 3

Задача 4. Волшебные комнаты

Источник:	основная
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Рассмотрим лабиринт, состоящий из комнат. Между любыми двумя комнатами изначально может быть коридор (но его может и не быть). Необходимо по коридорам как можно быстрее добраться из комнаты номер S в комнату номер F.

Однако лабиринт не простой, в нём также существуют *волшебные* комнаты: попадая в такую комнату можно совершить (но можно и не совершать) немного магии: все коридоры, которые существовали до этого, исчезают, однако если между двумя комнатами раньше коридора не существовало, то он появляется.

Формат входных данных

Первая строка входного файла содержит два числа N и M — количество комнат и количество коридоров соответственно ($1 \leqslant N, M \leqslant 10^5$).

Вторая строка содержит два числа S и F — соответственно номер начальной и конечной комнат ($1 \leqslant S, F \leqslant N$).

Третья строка содержит число W ($0 \leqslant W \leqslant N$) — количество волшебных комнат.

В четвёртой строке записано W чисел o_i, где o_i — номер i-ой волшебной комнаты. Все o_i различны.

В следующих M строках находится описание коридоров: i-ая строка содержит два числа v_i и u_i — номера комнат, которые соединяет i-ый коридор ($1 \leqslant v_i < u_i \leqslant N$). Между любыми двумя комнатами существет не более одного коридора.

Формат выходных данных

В выходной файл необходимо вывести одно целое число — минимальное количество коридоров, которое придётся преодолеть, чтобы добраться из комнаты номер S в комнату номер F.

Если из команты S невозможно попасть в комнату F, то нужно вывести слово `IMPOSSIBLE`.

Пример

input.txt	output.txt
5 5 1 5 1 2 1 2 1 3 1 4 3 5 4 5	2

Задача 5. Поиск пути

Источник:	повышенной сложности
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Создан робот для мытья полов. Требуется написать для него программное обеспечение. План комнаты представляется клеточным полем размера N на M. Некоторые клетки свободны для проезда, по остальным ездить запрещено (препятствия). Робот может за одну секунду сделать одно из действий: повернуться на 90 градусов в любую сторону или проехать вперёд на следующую клетку. Выезжать за пределы поля запрещается.

От вас просят написать модуль поиска пути. Робот изначально находится в некоторой клетке, повёрнут вверх. Требуется попасть в некоторую другую клетку за минимальное время.

Формат входных данных

В первой строке входного файла записаны два целых числа N, M — количество строк и столбцов на плане соответственно ($1 \leqslant N, M \leqslant 500$).

Следующие N строк содержат по M символов каждая — это план комнаты. Свободная клетка обозначается точкой, препятствие обозначается решёткой, начальная вершина символом S, а конечная — символом T.

Формат выходных данных

Если пути нет, то в выходной файл нужно вывести одно число -1.

В противном случае в первую строку требуется вывести минимальное время T. В следующую строку нужно выдать искомый путь, включая повороты. Движение вперёд обозначается символом F, поворот влево — символом L, а поворот вправо символом R.

Пример

input.txt	output.txt
6 8 ######## #.S#T..# #..#...# #..###.# #......# ########	17 RRFFFLFFFFLFFFLFF

Задача 6. Дорожки

Источник:	повышенной сложности
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

За последние 50 лет вокруг домов Новосибирского Академгородка протоптано великое множество тропинок и дорожек. Между двумя домами может быть даже несколько различных тропинок. Нет только дорожек, которые начинаются около какого-то дома, около него и заканчиваются.

Администрация городка решила навести порядок в этом месте: ненужные тропинки засадить газоном, оставшиеся дорожки заасфальтировать.

Руководители Академгородка объявили конкурс на создание проекта по асфальтированию дорожек. В проект должен войти только такой набор дорожек, который будет удовлетворять следующим требованиям:

- если можно было раньше добраться по тропинкам от одного дома до другого, то такая бы возможность оставалась;
- суммарная длина тропинок должна быть минимальной, так как стоимость асфальтирования напрямую зависит от длины дорожек.

Но оказалось, что с помощью разного набора дорожек, удовлетворяющих изложенным требованиям, можно составить минимальный по стоимости проект. Тогда администрация решила все дорожки, которые могут входить в один из таких проектов, пока оставить, а остальные распахать под газон.

Ваша задача: определить все дорожки, каждая из которых может входить хотя бы в один из проектов.

Формат входных данных

В первой строке входного файла записаны числа N и M — количество домов и количество дорожек ($1 \leqslant N \leqslant 10^6, 0 \leqslant M \leqslant 2 \cdot 10^5$). Дома занумерованы числами от 1 до N, дорожки — числами от 1 до M.

В следующих M строках даны описания дорожек по порядку номеров в формате — A B W, где A и B — номера домов, которые соединяет дорожка, W — ее длина ($A \neq B, -10^9 \leqslant W \leqslant 10^9$).

Формат выходных данных

В первую строку выходного файла необходимо вывести одно целое число — количество дорожек, которые могут входить в проект администрации. В следующей строке должны быть записаны через пробел K чисел — номера этих дорожек в порядке возрастания.

Пример

input.txt	output.txt
4 5 1 2 1 2 3 1 3 4 1 4 1 1 1 3 2	4 1 2 3 4