Задача A. Остовное дерево

Имя входного файла:	`spantree.in`
Имя выходного файла:	`spantree.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Даны точки на плоскости, являющиеся вершинами полного графа. Вес ребра равен расстоянию между точками, соответствующими концам этого ребра. Требуется в этом графе найти остовное дерево минимального веса.

Формат входного файла

Первая строка входного файла содержит натуральное число n — количество вершин графа ($1 \le n \le 5000$). Каждая из следующих n строк содержит два целых числа x_i, y_i — координаты i-й вершины ($-10\,000 \le x_i, y_i \le 10\,000$). Никакие две точки не совпадают.

Формат выходного файла

Первая строка выходного файла должна содержать одно вещественное число — вес минимального остовного дерева.

Примеры

spantree.in	spantree.out
3 0 0 1 0 0 1	2

Задача B. Остовное дерево 2

Имя входного файла:	`spantree2.in`
Имя выходного файла:	`spantree2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Требуется найти в связном графе остовное дерево минимального веса.

Формат входного файла

Первая строка входного файла содержит два натуральных числа n и m — количество вершин и ребер графа соответственно. Следующие m строк содержат описание ребер по одному на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — номера концов ребра и его вес соответственно ($1 \le b_i, e_i \le n$, $0 \le w_i \le 100\,000$). $n \le 20\,000, m \le 100\,000$.

Граф является связным.

Формат выходного файла

Первая строка выходного файла должна содержать одно натуральное число — вес минимального остовного дерева.

Примеры

spantree2.in	spantree2.out
4 4 1 2 1 2 3 2 3 4 5 4 1 4	7

Задача C. Транспортная сеть

Имя входного файла:	`transport.in`
Имя выходного файла:	`transport.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Вовочка только что был назначен министром транспорта. В стране в настоящее время отсутствуют какие-либо транспортные средства между своими городами, так что Вовочке нужно все построить с нуля. Для этого он может строить дороги, чтобы соединить пары городов, а также может строить аэропорты в городах.

Построить дорогу длиной l стоит $R \times l$. Дорогу можно строить только от одного города до другого, по прямой. Построить аэропорт в городе стоит A рублей.

Вовочка хочет, чтобы из каждого города можно было попасть в каждый. Формально, в город Y можно добраться из города X, если выполняется одно какое-либо из следующих условий:

- Существует прямая дорога между X и Y.
- В X и Y есть аэропорты.
- Существует город Z, такой, что Z достижим из A и Y достижим из Z.

По данным координатам городов и константам R и A, найдите минимальную стоимость транспортной сети, которую министерство транспорта может построить.

Формат входного файла

Первая строка входного файла содержит число n — число городов ($1 \le n \le 150$). Вторая строка содержит n чисел — x-координаты городов. Третья строка содержит n чисел — y-координаты городов. Координаты от 0 до 10^6. Четвертая строка содержит вещественные числа R и A.

Формат выходного файла

Выведите минимальную стоимость, которую можнно получить, с точностью до 6 знаков.

Примеры

transport.in	transport.out
4 0 0 400 400 0 100 0 100 1.0 150.0	500.0
5 0 0 400 400 2000 0 100 0 100 2000 1.0 500.0	1600.0
8 0 100 200 300 400 2000 2100 2200 0 100 200 300 400 2000 2100 2200 0.5 200.0	824.2640687119285

Задача D. Кратчайший путь в невзвешенном графе

Имя входного файла:	`pathbge1.in`
Имя выходного файла:	`pathbge1.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан неориентированный невзвешенный граф. Найдите кратчайшее расстояние от первой вершины до всех вершин.

Формат входного файла

В первой строке входного файла два числа: n и m ($2 \le n \le 30000, 1 \le m \le 400000$), где n — количество вершин графа, а m — количество ребер.

Следующие m строк содержат описание ребер. Каждое ребро задается стартовой вершиной и конечной вершиной. Вершины нумеруются с единицы.

Формат выходного файла

Выведите n чисел — для каждой вершины кратчайшее расстояние до нее.

Пример

pathbge1.in	pathbge1.out
2 1 2 1	0 1

Задача E. Поиск цикла

Имя входного файла:	`cycle.in`
Имя выходного файла:	`cycle.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный невзвешенный граф. Необходимо определить есть ли в нём циклы, и если есть, то вывести любой из них.

Формат входного файла

В первой строке входного файла находятся два натуральных числа N и M ($1 \le N \le 100\,000, M \le 100\,000$) — количество вершин и рёбер в графе соответственно. Далее в M строках перечислены рёбра графа. Каждое ребро задаётся парой чисел — номерами начальной и конечной вершин соответственно.

Формат выходного файла

Если в графе нет цикла, то вывести «NO», иначе — «YES» и затем перечислить все вершины в порядке обхода цикла.

Пример

cycle.in	cycle.out
2 2 1 2 2 1	YES 2 1
2 2 1 2 1 2	NO

Задача F. Топологическая сортировка

Имя входного файла:	`topsort.in`
Имя выходного файла:	`topsort.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный невзвешенный граф. Необходимо его топологически отсортировать.

Формат входного файла

В первой строке входного файла даны два натуральных числа N и M ($1 \le N \le 100\,000$, $0 \le M \le 100\,000$) — количество вершин и рёбер в графе соответственно. Далее в M строках перечислены рёбра графа. Каждое ребро задаётся парой чисел — номерами начальной и конечной вершин соответственно.

Формат выходного файла

Вывести любую топологическую сортировку графа в виде последовательности номеров вершин. Если граф невозможно топологически отсортировать, вывести -1.

Пример

topsort.in	topsort.out
6 6 1 2 3 2 4 2 2 5 6 5 4 6	4 6 3 1 2 5
3 3 1 2 2 3 3 1	-1

Задача G. Кратчайший путь

Имя входного файла:	`shortpath.in`
Имя выходного файла:	`shortpath.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Дан ориентированный взвешенный ациклический граф. Требуется найти в нем кратчайший путь из вершины s в вершину t.

Формат входного файла

Первая строка входного файла содержит четыре целых числа n, m, s и t — количество вершин, дуг графа, начальная и конечная вершина соответственно. Следующие m строк содержат описания дуг по одной на строке. Ребро номер i описывается тремя натуральными числами b_i, e_i и w_i — началом, концом и длиной дуги соответственно ($1 \le b_i, e_i \le n$, $|w_i| \le 1000$).

Входной граф не содержит циклов и петель.

$1 \le n \le 100\,000$, $0 \le m \le 200\,000$.

Формат выходного файла

Первая строка выходного файла должна содержать одно целое число — длину кратчайшего пути из s в t. Если пути из s в t не существует, выведите «Unreachable».

Пример

shortpath.in	shortpath.out
2 1 1 2 1 2 -10	-10
2 1 2 1 1 2 -10	Unreachable

Задача H. Гамильтонов путь

Имя входного файла:	`hamiltonian.in`
Имя выходного файла:	`hamiltonian.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный граф без циклов. Требуется проверить, существует ли в нем путь, проходящий по всем вершинам.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m — количество вершин и дуг графа соответственно. Следующие m строк содержат описания дуг по одной на строке. Ребро номер i описывается двумя натуральными числами b_i и e_i — началом и концом дуги соответственно ($1 \le b_i, e_i \le n$).

Входной граф не содержит циклов и петель.

$1 \le n \le 100\,000$, $0 \le m \le 200\,000$.

Формат выходного файла

Если граф удовлетворяет требуемому условию, то выведите `YES`, иначе `NO`.

Пример

hamiltonian.in	hamiltonian.out
3 3 1 2 1 3 2 3	YES
3 2 1 2 1 3	NO

Задача I. Игра

Имя входного файла:	`game.in`
Имя выходного файла:	`game.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан ориентированный невзвешенный ациклический граф. На одной из вершин графа стоит «фишка». Двое играют в игру. Пусть «фишка» находится в вершине u, и в графе есть ребро (u, v). Тогда за ход разрешается перевести «фишку» из вершины u в вершину v. Проигрывает тот, кто не может сделать ход.

Формат входного файла

В первой строке входного файла находятся три натуральных числа N, M и S ($1 \le N, S, M \le 100\,000$) — количество вершин рёбер и вершина, в которой находится «фишка» в начале игры соответственно. Далее в M строках перечислены рёбра графа. Каждое ребро задаётся парой чисел — номерами начальной и конечной вершин.

Формат выходного файла

Если выигрывает игрок, который ходит первым, выведите «First player wins», иначе — «Second player wins».

Пример

game.in	game.out
3 3 1 1 2 2 3 1 3	First player wins
3 2 1 1 2 2 3	Second player wins