

Задача А. От списка ребер к матрице смежности

Имя входного файла: e2m.in
Имя выходного файла: e2m.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Простой неориентированный граф задан списком ребер, выведите его представление в виде матрицы смежности.

Формат входного файла

Входной файл содержит числа N ($1 \leq N \leq 100$) — число вершин в графе и M ($1 \leq M \leq \frac{n(n-1)}{2}$) — число ребер. Затем следует M пар чисел — ребра графа.

Формат выходного файла

Выведите в выходной файл матрицу смежности заданного графа.

Пример

e2m.in	e2m.out
3 3	0 1 1
1 2	1 0 1
2 3	1 1 0
1 3	

Задача В. От матрицы смежности к списку ребер

Имя входного файла: `m2e.in`
Имя выходного файла: `m2e.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Простой неориентированный граф задан матрицей смежности, выведите его представление в виде списка ребер.

Формат входного файла

Входной файл содержит число N ($1 \leq N \leq 100$) — число вершин в графе, и затем N строк по N чисел, каждое из которых равно 0 или 1 — его матрицу смежности.

Формат выходного файла

Выведите в выходной файл список ребер заданного графа. Ребра можно выводить в произвольном порядке.

Пример

<code>m2e.in</code>	<code>m2e.out</code>
3	1 2
0 1 1	2 3
1 0 1	1 3
1 1 0	

Задача C. Lines

Имя входного файла: `lines.in`
Имя выходного файла: `lines.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

В таблице $N \times N$ столбцов некоторые клетки заняты шариками, другие свободны. Выбран шарик, который нужно переместить, и место, куда его нужно переместить. Выбранный шарик за один шаг перемещается в соседнюю по горизонтали или вертикали свободную клетку. Требуется выяснить, возможно ли переместить шарик из начальной клетки в заданную, и если возможно, то найти путь из наименьшего количества шагов.

Формат входного файла

В первой строке входного файла находится число N ($1 < N \leq 250$), в следующих N строках – по N символов. Символом точки обозначена свободная клетка, латинской заглавной `O` – шарик, `@` – исходное положение шарика, который должен двигаться, латинской заглавной `X` – конечное положение шарика.

Формат выходного файла

В первой строке выходного файла выводится `Y`, если движение возможно, или `N`, если нет. Если движение возможно, далее следует N строк по N символов – как и на вводе, но `X`, а также все точки по пути заменяются плюсами `+`.

Пример

lines.in	lines.out
5 ...X. O.OOO@	Y ..++. ..++.. O+OOO ..++++@
5 ..X.. OOOOO@..	N

Задача D. Shortest Path

Имя входного файла: `path.in`
Имя выходного файла: `path.out`
Ограничение по времени: 2 seconds
Ограничение по памяти: 64 megabytes

You are given a weighed directed graph and a vertex s in it. For each vertex u find the length of the shortest path from s to u .

Формат входного файла

The first line of the input file contains n , m and s — the number of vertices and edges in the graph, and the number of the starting vertex, respectively ($2 \leq n \leq 2\,000$, $1 \leq m \leq 5\,000$).

The following m lines describe edges. Each edge is specified with its start vertex, its end vertex, and its weight. The weight of any edge is integer and does not exceed 10^{15} by its absolute value. There can be several edges between a pair of vertices. There can be an edge between a vertex and itself.

Формат выходного файла

Output n lines — for each vertex u output the length of the shortest path from s to u , '*' if there is no path from s to u , or '-' if there is no shortest path from s to u .

Пример

path.in	path.out
6 7 1	0
1 2 10	10
2 3 5	-
1 3 100	-
3 5 7	-
5 4 10	*
4 3 -18	
6 1 -1	

Задача Е. Дейкстра

Имя входного файла: `dijkstra.in`
Имя выходного файла: `dijkstra.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Дан ориентированный взвешенный граф. Найдите кратчайшее расстояние от одной заданной вершины до другой.

Формат входного файла

В первой строке входного файла три числа: N , S и F ($1 \leq N \leq 2000$, $1 \leq S, F \leq N$), где N — количество вершин графа, S — начальная вершина, а F — конечная. В следующих N строках по N чисел — матрица смежности графа, где -1 означает отсутствие ребра между вершинами, а любой неотрицательное число — присутствие ребра данного веса. На главной диагонали матрицы всегда нули.

Формат выходного файла

Вывести искомое расстояние или -1 , если пути между указанными вершинами не существует.

Пример

dijkstra.in	dijkstra.out
3 1 2 0 -1 2 3 0 -1 -1 4 0	6

Задача F. Заправки

Имя входного файла: petrol.in
Имя выходного файла: petrol.out
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

В стране N городов, некоторые из которых соединены между собой дорогами. Для того, чтобы проехать по одной дороге, требуется один бак бензина. В каждом городе бак бензина имеет разную стоимость. Вам требуется добраться из первого города в N -ый, потратив как можно меньшее количество денег.

Формат входного файла

Во входном файле записано сначала число N ($1 \leq N \leq 100$), затем идёт N чисел, i -е из которых задаёт стоимость бензина в i -м городе (всё это целые числа из диапазона от 0 до 100). Затем идёт число M — количество дорог в стране, далее идёт описание самих дорог. Каждая дорога задаётся двумя числами — номерами городов, которые она соединяет. Все дороги двусторонние (то есть, по ним можно ездить как в одну, так и в другую сторону), между двумя городами всегда существует не более одной дороги, не существует дорог, ведущих из города в себя.

Формат выходного файла

В выходной файл выведите одно число — суммарную стоимость маршрута или -1 , если добраться невозможно.

Пример

petrol.in	petrol.out
4 1 10 2 15 4 1 2 1 3 4 2 4 3	3
4 1 10 2 15 0	-1

Задача G. Флойд

Имя входного файла: `floyd.in`
Имя выходного файла: `floyd.out`
Ограничение по времени: 1 секунда
Ограничение по памяти: 64 мегабайта

Полный ориентированный взвешенный граф задан матрицей смежности. Постройте матрицу кратчайших путей между его вершинами. Гарантируется, что в графе нет циклов отрицательного веса.

Формат входного файла

В первой строке вводится единственное число N ($1 \leq N \leq 100$) — количество вершин графа. В следующих N строках по N чисел задаётся матрица смежности графа (j -е число в i -й строке соответствует весу ребра из вершины i в вершину j). Все числа по модулю не превышают 100. На главной диагонали матрицы — всегда нули.

Формат выходного файла

Выведите N строк по N чисел — матрицу кратчайших расстояний между парами вершин. j -е число в i -й строке должно быть равно весу кратчайшего пути из вершины i в вершину j .

Пример

floyd.in	floyd.out
4	0 6 7 13
0 9 100	12 0 2 8
100 0 2 8	11 16 0 7
100 100 0 7	4 9 11 0
4 100 100 04	

Задача Н. Pink Floyd

Имя входного файла:	floyd.in
Имя выходного файла:	floyd.out
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайта

Группа Pink Floyd собирается дать новый концертный тур по всему миру. По предыдущему опыту группа знает, что солист Роджер Уотерс постоянно нервничает при перелетах. На некоторых маршрутах он теряет вес от волнения, а на других — много ест и набирает вес.

Известно, что чем больше весит Роджер, тем лучше выступает группа, поэтому требуется спланировать перелеты так, чтобы вес Роджера на каждом концерте был максимально возможным.

Группа должна посещать города в том же порядке, в котором она дает концерты. При этом между концертами группа может посещать промежуточные города.

Формат входного файла

Первая строка входного файла содержит три натуральных числа n , m и k — количество городов в мире, количество рейсов и количество концертов, которые должна дать группа соответственно ($n \leq 100$, $m \leq 10\,000$, $2 \leq k \leq 10\,000$). Города пронумерованы числами от 1 до n .

Следующие m строк содержат описание рейсов, по одному на строке. Рейс номер i описывается тремя числами b_i , e_i и w_i — номер начального и конечного города рейса и предполагаемое изменение веса Роджера в миллиграммах ($1 \leq b_i, e_i \leq n$, $-100\,000 \leq w_i \leq 100\,000$).

Последняя строка содержит числа a_1, a_2, \dots, a_k — номера городов, в которых проводятся концерты ($a_i \neq a_{i+1}$). В начале концертного тура группа находится в городе a_1 .

Гарантируется, что группа может дать все концерты.

Формат выходного файла

Первая строка выходного файла должна содержать число l — количество рейсов, которые должна сделать группа. Вторая строка должна содержать l чисел — номера используемых рейсов.

Если существует такая последовательность маршрутов между концертами, что Роджер будет набирать вес неограниченно, то первая строка выходного файла должна содержать строку “infinitely kind”.

Пример

floyd.in	floyd.out
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 -10 1 3 1 2 4	6 5 6 5 7 2 3
4 8 5 1 2 -2 2 3 3 3 4 -5 4 1 3 1 3 2 3 1 -2 3 2 -3 2 4 10 1 3 1 2 4	infinitely kind

Задача I. 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 ($1 \leq n \leq 5000$). Каждая из следующих n строк содержит два целых числа x_i, y_i — координаты i -й вершины ($-10\,000 \leq x_i, y_i \leq 10\,000$). 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 J. 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 ($1 \leq b_i, e_i \leq n$, $0 \leq w_i \leq 100\,000$). $n \leq 20\,000$, $m \leq 100\,000$.

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 К. Максимальный поток

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

адан ориентированный граф, каждое ребро которого обладает целочисленной пропускной способностью. Найдите максимальный поток из вершины с номером 1 в вершину с номером n .

Формат входного файла

Первая строка входного файла содержит n и m — количество вершин и количество ребер графа ($2 \leq n \leq 100$, $1 \leq m \leq 1000$). Следующие m строк содержат по три числа: номера вершин, которые соединяет соответствующее ребро графа и его пропускную способность. Пропускные способности не превосходят 10^5 .

Формат выходного файла

В выходной файл выведите одно число — величину максимального потока из вершины с номером 1 в вершину с номером n .

Примеры

maxflow.in	maxflow.out
4 5 1 2 1 1 3 2 3 2 1 2 4 2 3 4 1	3

Задача L. Максимальный поток минимальной стоимости

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

адан ориентированный граф, каждое ребро которого обладает пропускной способностью и стоимостью. Найдите максимальный поток минимальной стоимости из вершины с номером 1 в вершину с номером n .

Формат входного файла

Первая строка входного файла содержит n и m — количество вершин и количество ребер графа ($2 \leq n \leq 100$, $1 \leq m \leq 1000$). Следующие m строк содержат по четыре целых числа: номера вершин, которые соединяет соответствующее ребро графа, его пропускную способность и его стоимость. Пропускные способности и стоимости не превосходят 10^5 .

Формат выходного файла

В выходной файл выведите одно число — цену максимального потока минимальной стоимости из вершины с номером 1 в вершину с номером n . Ответ не превышает $2^{63} - 1$. Гарантируется, что в графе нет циклов отрицательной стоимости.

Примеры

<code>mincost.in</code>	<code>mincost.out</code>
4 5 1 2 1 2 1 3 2 2 3 2 1 1 2 4 2 1 3 4 2 3	12

Задача М. Криптография

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

дано n матриц A_1, A_2, \dots, A_n размера 2×2 . Необходимо для нескольких запросов вычислить произведение матриц A_i, A_{i+1}, \dots, A_j . Все вычисления производятся по модулю r .

Формат входного файла

Первая строка входного файла содержит числа r ($1 \leq r \leq 10\,000$), n ($1 \leq n \leq 200\,000$) и m ($1 \leq m \leq 200\,000$). Следующие n блоков по две строки содержащие по два числа в строке — описания матриц. Затем следуют m пар целых чисел от 1 до n , запросы на произведение на отрезке.

Формат выходного файла

Выведите m блоков по две строки, по два числа в каждой — произведения на отрезках. Разделяйте блоки пустой строкой. Все вычисления производятся по модулю r .

Пример

crypto.in	crypto.out
3 4 4	0 2
0 1	0 0
0 0	
	0 2
2 1	0 1
1 2	
	0 1
0 0	0 0
0 2	
	2 1
1 0	1 2
0 2	
1 4	
2 3	
1 3	
2 2	

Задача N. RMQ

Имя входного файла: `rmq2.in`
Имя выходного файла: `rmq2.out`
Ограничение по времени: 2 секунды
Ограничение по памяти: 256 мегабайта

В первой строке находится число n — размер массива. ($1 \leq n \leq 100000$) Во второй строке находится n чисел a_i — элементы массива. Далее содержится описание операций, их количество не превышает 200000. В каждой строке находится одна из следующих операций:

- **set** $i\ j\ x$ — установить все $a[k]$, $i \leq k \leq j$ в x .
- **add** $i\ j\ x$ — увеличить все $a[k]$, $i \leq k \leq j$ на x .
- **min** $i\ j$ — вывести значение минимального элемента в массиве на отрезке с i по j , гарантируется, что $(1 \leq i \leq j \leq n)$.

Все числа во входном файле и результаты выполнения всех операций не превышают по модулю 10^{18}

Формат выходного файла

Выведите последовательно результат выполнения всех операций **min**. Следуйте формату выходного файла из примера.

Пример

rmq2.in	rmq2.out
5	2
1 2 3 4 5	1
min 2 5	1
min 1 5	2
min 1 4	5
min 2 4	5
set 1 3 10	8
add 2 4 4	8
min 2 5	
min 1 5	
min 1 4	
min 2 4	

Задача О. Переместить в начало

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

Вам дан массив $a_1 = 1, a_2 = 2, \dots, a_n = n$ и последовательность операций: переместить элементы с l_i по r_i в начало массива. Например, для массива 2, 3, 6, 1, 5, 4, после операции (2, 4) новый порядок будет 3, 6, 1, 2, 5, 4. А после применения операции (3, 4) порядок элементов в массиве будет 1, 2, 3, 6, 5, 4.

Выведите порядок элементов в массиве после выполнения всех операций.

Формат входного файла

В первой строке входного файла указаны числа n и m ($2 \leq n \leq 100\,000$, $1 \leq m \leq 100\,000$) — число элементов в массиве и число операций. Следующие m строк содержат операции в виде двух целых чисел: l_i и r_i ($1 \leq l_i \leq r_i \leq n$).

Формат выходного файла

Выведите n целых чисел — порядок элементов в массиве после применения всех операций.

Пример

<code>movetofront.in</code>	<code>movetofront.out</code>
6 3 2 4 3 5 2 2	1 4 5 2 3 6