

Задача А. Лестница

Максимальное время работы на одном тесте: 1 секунда
Максимальный объем используемой памяти: 64 мегабайта

Археологи раскопали Древний Храм, ко входу в который ведет лестница, шириной в 1 (один) метр, из M ступенек различной длины и высоты. Лестница построена из каменных блоков $1 \times 1 \times 1$ метр. Археологи хотят для удобства туристов, чтобы лестница состояла из меньшего количества ступенек N . Для этого они могут также устанавливать каменные блоки $1 \times 1 \times 1$. Какое минимальное количество блоков необходимо, чтобы сделать лестницу в N ступенек, если известны начальная длина и высота каждой ступеньки. Высоты и длины ступенек новой лестницы могут различаться.

Формат входных данных

В первой строке через пробел заданы два целых числа M и N ($1 \leq N < M \leq 100$). Далее идут M строк, содержащих пару целых чисел L и H - длина и высота i -ой ступеньки соответственно ($1 \leq L, H \leq 101$). Ступеньки нумеруются снизу вверх.

Формат выходных данных

В выходной файл выведите единственное число - ответ на задачу.

Пример

Входные данные	Выходные данные
5 3 4 2 1 2 5 2 1 2 2 1	3

Задача В. Раскраска кубиков

Максимальное время работы на одном тесте: 2 секунды
Максимальный объем используемой памяти: 64 мегабайта

На день рождения Пете подарили коробку кубиков. На каждом кубике написано некоторое целое число. Петя выложил все n своих кубиков в ряд, так что числа на кубиках оказались расположены в некотором порядке $a[1], a[2], \dots, a[n]$. Теперь он хочет раскрасить кубики в разные цвета таким образом, чтобы для каждого цвета последовательность чисел на кубиках этого цвета была строго возрастающей. То есть, если кубики с номерами $i[1], i[2], \dots, i[k]$ покрашены в один цвет, то $a[i[1]] < a[i[2]] < \dots < a[i[k]]$. Петя хочет использовать как можно меньше цветов. Помогите ему!

Формат входных данных

Первая строка входного файла содержит число n - количество кубиков у Пети ($1 \leq n \leq 250000$). Затем следует n чисел, разделенных пробелами и/или переводами строки - $a[1], a[2], \dots, a[n]$

Формат выходных данных

На первой строке выходного файла выведите число L - наименьшее количество цветов, которое потребуется Пете. На следующей строке выведите n чисел из диапазона от 1 до L - цвета, в которые Петя должен покрасить кубики.

Пример

Входные данные	Выходные данные
5 1 2 4 3 5	2 1 1 1 2 2

Задача С. Лестницу в треугольник

Максимальное время работы на одном тесте: 1 секунда

Максимальный объем используемой памяти: 64 мегабайта

На плоскости дана геометрическая фигура "лестница". Она имеет N ступенек, которые заданы положительными координатами. Каждая ступень имеет свою высоту и ширину. Требуется найти прямую, которая отсекает от некоторых ступеней "лестницы" треугольники так, что из полученных фигур можно сложить прямоугольный треугольник такой же площади, что и исходная фигура. Разрешается, чтобы отсекаемые от ступеней треугольники соприкасались только вершинами (но не сторонами).

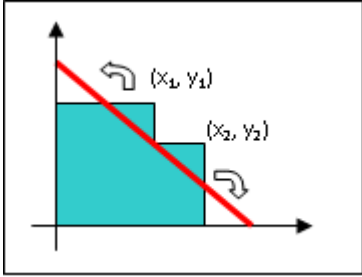
Формат входных данных

В первой строке дано число $0 \leq N \leq 1000$. Далее записаны N строк. Каждая строка содержит два целых числа через пробел $0 < x_i, y_i < 10^6$ - координаты вершины i -й ступени (ступени перечисляются в порядке сверху вниз, слева направо).

Формат выходных данных

Файл содержит одну строку: два числа через пробел - высота и ширина получившегося прямоугольного треугольника. Если существует несколько решений, то вывести любое. Результат выводится с точностью до четырех десятичных знаков после запятой. В случае, когда решение отсутствует, вывести два нуля через пробел

Примеры

Входные данные	Выходные данные
2 2 3 3 2 	4.0000 4.0000
2 1 3 3 1	0 0

Задача D. Тир

Максимальное время работы на одном тесте: 1 секунда
 Максимальный объем используемой памяти: 64 мегабайта

Шесть стрелков решили повысить свою меткость и зашли в тир пострелять. В тире было установлено шесть мишеней, и каждый стрелок выстрелил во все мишени. После чего они собрались в баре, и каждый рассказал, сколько раз он попал в мишень. Бармен, запомнил, что сказал каждый из них, и на следующий день посчитал, сколько дырок в каждой мишени. Требуется помочь бармену определить: не ошибся ли кто-то из "снайперов", и сколько существует вариантов стрельбы (то есть, кто в какие мишень попадал), при которых получаются такие результаты. Считается, что две пули в одну дырку не попадают.

Формат входных данных

Во входном файле в первой строке для каждого стрелка записано число попаданий, которое он назвал при обсуждении результатов. В третьей строке записано число дырок для каждой мишени

Формат выходных данных

В первой строке - количество вариантов такой стрельбы. Если вариант всего один, то для каждого стрелка вывести строчку, в которой будет для каждой мишени проставлено 0 - не попал, 1 - попал.

Примеры

Входные данные	Выходные данные
2 0 0 0 0 0 0 1 0 1 0 0	1 0 1 0 1 0
1 1 1 1 1 1 1 1 1 1 1 1	720

Задача Е. Последовательность

Максимальное время работы на одном тесте: 1 секунда
Максимальный объем используемой памяти: 64 мегабайта

Последовательность чисел монотонно убывающая если для любого i элемента (кроме последнего) выполняется $A_i > A_{i+1}$. Последовательность чисел монотонно возрастающая если для любого i элемента (кроме последнего) выполняется $A_i < A_{i+1}$. Последовательность монотонная если она или монотонно убывающая или монотонно возрастающая. Дана последовательность A_i из n чисел. Каждый элемент последовательности можно увеличивать или уменьшать. Требуется изменить последовательность так чтобы в ней содержалось K монотонных непересекающихся последовательностей. (один элемент тоже последовательность). Т.к. преобразований бесконечно много то нужно найти такое преобразование чтобы сумма разностей по модулю конечного значения A_i и начального значения A_i была минимальна.

Формат входных данных

В первой строчке n, k — количество элементов в последовательности, k сколько должно быть монотонных последовательностей. Во второй строчке n чисел через пробел — сама последовательность (каждое число по модулю меньше 20 000 000)

Формат выходных данных

Одно число, ответ на задачу

Примеры

Входные данные	Выходные данные
4 2 1 1 1 1	2
11 3 1 10 4 2 5 1 3 2 4 6 1	11

Задача F. План уничтожения

Максимальное время работы на одном тесте: 2 секунды
Максимальный объем используемой памяти: 64 мегабайта

Секретный бункер уходит на N этажей вниз. Под нижним этажом бункера находится сверхсекретная лаборатория. Злобный диверсант хочет вывести лабораторию из строя, залив её водой (даже очень небольшого количества воды хватит, чтобы запоганить сверхточные приборы). Для этого он использует лужицы воды, остающиеся от жизнедеятельности обитателей бункера. В лужицах i -го этажа находится E_i воды. Диверсанту известно, что если на нём скопится больше C_i воды, то перегородка не выдержит и вся вода сольется на этаж ниже. Он может проделать отверстия в некоторых перегородках, по которым вода также стечет вниз. Прodelать отверстие в полу i -го этажа стоит P_i у.е. Помогите диверсанту уничтожить лабораторию с минимальными материальными затратами.

Формат входных данных

Первая строка входного файла содержит натуральное число N ($1 \leq N \leq 50000$) - количество этажей в бункере, в следующих N строках находятся тройки целых чисел C_i, E_i, P_i ($0 < E_i \leq C_i < 1000000$; $E_1 + E_2 + \dots + E_N < 20000000000$; $P_i > 0$; $P_1 + P_2 + \dots + P_N < 20000000000$). Этажи нумеруются сверху вниз.

Формат выходных данных

В первой строке выходного файла выдать количество денег, которое придется потратить злобному диверсанту, в следующих строках выведите номера этажей, в полу которых следует проделать отверстия.

Пример

Входные данные	Выходные данные
4 1 1 1 1 1 3 3 1 2 3 1 10	3 1 3

Задача G. Вполоборота

Максимальное время работы на одном тесте: 1 секунда
Максимальный объем используемой памяти: 64 мегабайта

Словом считается непустая последовательность больших и маленьких латинских букв. Вам необходимо развернуть все слова во входном файле задом наперед.

Формат входных данных

Во входном файле записано все что угодно. Его размер не превышает 1 Мб.

Формат выходных данных

В выходном файле должен содержаться входной файл, в котором все слова развернуты. Все прочие символы, в том числе переводы строк, должны остаться нетронутыми

Пример

Входные данные	Выходные данные
Summer Informatics School. Now in Perm!	remmuS scitamrofni loohcS. woN ni mreP!

Задача Н. Просто матрица

Максимальное время работы на одном тесте: 1 секунды
 Максимальный объем используемой памяти: 64 мегабайта

В этой задаче не будет идти речь о Бердяндии, о дорогах или авиарейсах. Здесь не будет надоевших монет. Речь пойдет о простой квадратной матрице.

В квадратную матрицу A порядка n были вписаны числа от 1 до n^2 , каждое по одному разу. Затем для каждого числа была записана пара чисел $top_{i,j}$ и $left_{i,j}$, где $top_{i,j}$ это количество чисел в столбце j стоящих выше числа $A_{i,j}$ и одновременно больших него, а $left_{i,j}$ это количество чисел в строке i стоящих левее числа $A_{i,j}$ и одновременно больших него.

Вам заданы матрица top и $left$. Ваша задача состоит в нахождении возможной матрицы A , соответствующей входным данным.

Формат входных данных

В первой строке входного файла записано целое число n ($1 \leq n \leq 600$), где n это порядок искомой матрицы. Далее во входном файле записана пара матриц top и $left$, в виде n строк по n чисел в каждой строке. Матрицы разделены пустой строкой.

Формат выходных данных

В выходной файл выведите матрицу A в формате, аналогичном входным данным. Если решений несколько, то выведите любое. Если решения не существует, выведите единственное число 0.

Пример

Входные данные	Выходные данные
3	1 2 6
0 0 0	5 3 7
0 0 0	9 8 4
0 0 2	
0 0 0	
0 1 0	
0 1 2	

Задача I. Волки и овцы

Максимальное время работы на одном тесте: 2 секунды
Максимальный объем используемой памяти: 64 мегабайта

Пастбища Берляндии в опасности. Волки напали на пастбище овец. Пастух решил застрелить всех волков, при этом не убив ни одной овцы. Ружье заряжено бронебойными патронами, поэтому пули пролетают насквозь. Овцы и волки представлены отрезками. Пастух находится в точке $(0, 0)$. Траектория полета пули — луч, выходящий из точки $(0, 0)$. Если траектория пули имеет общую точку с отрезком, характеризующим животное, то животное умирает. Найдите наименьшее количество выстрелов, необходимое для убийства всех волков. Овцы при этом должны остаться живы.

Формат входных данных

В первой строке записаны два целых числа N и M ($0 \leq N \leq 10^5$, $0 \leq M \leq 10^5$) — количество волков и овец соответственно. Далее следует $N+M$ строк. В каждой строке записано четыре числа $X1$, $Y1$, $X2$, $Y2$ ($-1000 \leq X1, X2 \leq 1000$, $1 \leq Y1, Y2 \leq 1000$), описывающие отрезки. Первые N отрезков описывают положение волков, следующие M строк положение овец.

Формат выходных данных

Выведите наименьшее количество выстрелов, необходимое для убийства всех волков. Если невозможно убить всех волков, сохранив овец живыми, то выведите "No solution".

Примеры

Входные данные	Выходные данные
1 1 5 5 6 7 3 5 8 5	No solution
2 1 1 1 2 3 -5 4 2 2 999 1000 1000 999	1

Задача J. Вслепую по лабиринту

Максимальное время работы на одном тесте:

1 секунда

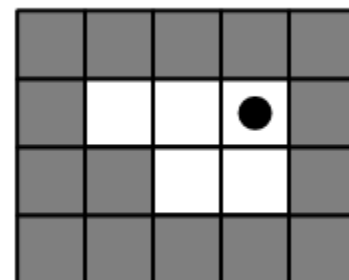
Максимальный объем используемой памяти:

64 мегабайт

Это интерактивная задача.

Ваша цель — написать программу, управляющую роботом, идущим вслепую по лабиринту.

Лабиринт состоит из N на M ($1 \leq N, M \leq 30$) клеток. Каждая из клеток может быть свободной или заблокированной (непроходимой). Все клетки на границе лабиринта непроходимые. Робот начинает работу в свободной клетке, он может переместиться на юг, запад, север или восток свободную клетку. Робот не имеет оптических сенсоров, только сенсор удара, так что при попытке перемещения в заблокированную клетку сработает сенсор и робот останется в той же клетке.



В

Робот должен побывать во всех проходимых клетках лабиринта.

Из начальной клетки гарантированно можно попасть во все достижимые клетки лабиринта.

Протокол интерактивного взаимодействия

Программа должна выводить на стандартный вывод одну строку с действием робота и ждать строки в стандартном вводе с ответом, затем выводить очередную строку с действием и считывать ответ и так далее до тех пор, пока все проходимые клетки лабиринта не будут посещены. Программа должна завершать работу только когда все проходимые клетки будут посещены. Проходимые клетки могут быть посещены несколько раз. Допустимо передвигаться даже если все проходимые клетки уже посещены.

Формат выходных данных

Каждая строка выходных данных должна представлять собой команду для робота. Это должна быть одна из пяти возможных строк: SOUTH, WEST, NORTH, EAST, или DONE. Строка DONE должна быть напечатана после посещения роботом всех проходимых клеток. После вывода DONE программа должна завершать свою работу. Необходимо очищать выходной буфер после вывода каждой команды (flush).

Формат входных данных

Каждая строка входных данных представляет собой ответ на действие робота. Это может быть строка EMPTY если робот успешно переместился в заданном направлении или строка BLOCKED если робот не смог переместиться из-за того, что клетка, в которую он хотел попасть, непроходима.

Примеры

Входные данные	Выходные данные
NORTH	BLOCKED
EAST	BLOCKED
SOUTH	EMPTY
EAST	BLOCKED
SOUTH	BLOCKED
WEST	EMPTY
SOUTH	BLOCKED
WEST	BLOCKED
NORTH	EMPTY
WEST	EMPTY
WEST	BLOCKED
NORTH	BLOCKED
EAST	EMPTY
NORTH	BLOCKED
DONE	

Задача К. Дерево?

Максимальное время работы на одном тесте: 2 секунды

Максимальный объем используемой памяти: 64 мегабайта

Имеется неориентированный граф, состоящий из N вершин и M ребер. Необходимо проверить, является ли граф деревом. Напомним, что дерево — это связный граф, в котором нет циклов (следовательно, между любой парой вершин существует ровно один простой путь). Граф называется связным, если от одной вершины существует путь до любой другой.

Формат входных данных

Во входном файле в первой строке содержатся два целых числа N и M ($1 \leq N \leq 100$, $0 \leq M \leq 1000$), записанные через пробел. Далее следуют M различных строк с описаниями ребер, каждая из которых содержит два натуральных числа A_i и B_i ($1 \leq A_i < B_i \leq N$), где A_i и B_i — номера вершин, соединенных i -м ребром.

Формат выходных данных

В выходной файл выведите слово YES, если граф является деревом или NO в противном случае.

Пример

Входные данные	Выходные данные
3 2 1 2 1 3	YES