Задача A. Древние цивилизации

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

Недавно Петя занялся изучением древних цивилизаций. Он нашел в энциклопедии даты рождения и гибели N различных древних цивилизаций и теперь хочет узнать о влиянии культуры одних цивилизаций на культуру других.

Петя предположил, что между цивилизациями A и B происходил культурный обмен, если они сосуществовали в течении некоторого ненулевого промежутка времени. Например, если цивилизация A зародилась в 600 году до н.э. и существовала до 400 года до н.э., а цивилизация B зародилась в 450 году до н.э. и существовала до 300 года до н.э., то культура каждой из этих цивилизаций оказывала влияние на развитие другой цивилизации в течении 50 лет. В то же время, если цивилизация C зародилась в 400 году до н.э. и существовала до 50 года до н.э., то она не смогла осуществить культурного обмена с цивилизацией A, в то время как культурный обмен с цивилизацией B продолжался в течении 100 лет.

Теперь для выполнения своих исследований Петя хочет найти такую пару цивилизаций, культурный обмен между которыми имел место на протяжении наименьшего ненулевого промежутка времени. Помогите ему!

Формат входных данных

Первая строка входного файла содержит число N — количество цивилизаций, культура которых интересует Петю ($1 \le N \le 100\,000$). Следующие N строк содержат описание цивилизаций — каждая строка содержит пару целых чисел S_i и E_i — год зарождения и год гибели соответствующей цивилизации. Все числа не превосходят 10^9 по абсолютной величине, $S_i < E_i$.

Формат выходных данных

Выведите два числа — номера цивилизаций, периоды существования которых имеют наименьшее ненулевое пересечение. Если никакие две цивилизации не пересекаются во времени, выведите в выходной файл единственное число 0.

Пример

ancient.in	ancient.out
3 -600 -400 -450 -300 -400 -50	1 2
2 -10 0 0 10	0
3 -10 80 -80 10 60 90	1 3

Задача B. Дробь

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

Известно, что сложение и умножение являются ассоциативными операциями. Это значит, что значение выражений вида $a_1 + a_2 + \ldots + a_n$ и $a_1 \cdot a_2 \cdot \ldots \cdot a_n$ не зависит от порядка выполнения в них действий и следовательно не меняется при произвольной расстановке в этих выражениях скобок.

В отличии от сложения и умножения, деление — операция не ассоциативная. Так, значение выражения вида $a_1/a_2/\cdots/a_n$ может меняться при расстановке в нем скобок.

Рассмотрим выражение вида

$$p_1/p_2/\cdots/p_n,$$

где все p_i — простые числа (не обязательно различные). Найдите количество возможных значений, которые может принять указанное выражение после расстановки в нем скобок, а также количество целых чисел среди этих значений.

Например, выражение $3/2/2$ после расстановки скобок может принять два значения: $3/4 = (3/2)/2$ и $3 = 3/(2/2)$.

Формат входных данных

Первая строка входного файла содержит число n ($1 \le n \le 200$). Следующая строка содержат n натуральных чисел — $p_1, p_2, \ldots, p_n$. Все числа p_i простые и не превосходят 10^4.

Формат выходных данных

На первой строке выходного файла выведите количество возможных значений, которые может принять выражение $p_1/p_2/\cdots/p_n$ при заданных p_i после расстановки в нем скобок. На второй строке выведите количество целых чисел среди этих значений.

Пример

fraction.in	fraction.out
3 3 2 2	2 1

Задача C. Захват королевства

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

В одном магическом королевстве есть N городов, каждые два из которых соединены дорогой. Эти дороги были построены в давние времена Светлыми и Темными силами, те дороги, которые были построены Светлыми силами вымощены белыми камнями, а те, которые построены Темными — черными. Поскольку магические чары охраняют дороги, ни одно доброе существо не может пройти по дороге, вымощенной черными камнями, и ни одно злое — по белой дороге.

Когда-то давно люди решили избрать своих правителей и изгнали верховных магов из королевства. Однако недавно верховные маги Светлых и Темных сил договорились и решили вернуть королевство под свой контроль. Для этого они хотят направить в некоторые города королевства магов, которые возьмут эти и смежные города под свой контроль.

Точнее, если светлый маг будет направлен в некоторый город, то он возьмет под свой контроль этот город и все города, которые напрямую соединены с ним белыми дорогами. Аналогично, черный маг помимо города, в который он направлен, будет контролировать все города, напрямую соединенные с ним черными дорогами. Для захвата королевства требуется установить контроль над всеми городами.

Однако при разработке плана захвата обнаружилось две трудности. Во-первых, выяснилось, что маг согласен принять участие в операции только если все маги, которые будут направлены в королевство будут представлять ту же силу, что и он. То есть либо все участвующие в захвате маги должны быть светлыми, либо все они должны быть темными. Во-вторых, общее число магов, которые могут быть направлены в королевство не должно превышать K. Единственная надежда верховных магов заключается в том, что K достаточно велико, $2^K \geq N$.

Выясните, светлых или темных магов следует использовать для захвата королевства, а также в какие города их следует направить.

Формат входных данных

Первая строка входного файла содержит целые числа N и K ($2 \leq N \leq 256$, $2^K \geq N$, $K \leq N$).

Следующие N строк содержат по N целых чисел каждая. На i-ой позиции i-ой из этих строк расположено число 0, которое означает, что город не соединен дорогой сам с собой. Для всех $j \neq i$ число на j-ой позиции i-ой из этих строк равно 1, если i-ый город соединен с j-ым белой дорогой и равно 2, если они соединены черной дорогой. Числа в строках разделены пробелами.

Гарантируется, что входные данные корректны, то есть если i-ый город соединен с j-ым белой дорогой, то и j-ый соединен с i-ым белой дорогой, аналогично в случае черных дорог.

Формат выходных данных

Если захватить королевство при заданных условиях невозможно, выведите в выходной файл единственное число 0.

В противном случае на первой строке выведите 1, если удастся захватить королевство с использованием светлых магов, и 2, если требуется использовать черных магов. На следующей строке выведите число $L \leq K$ — количество использованных магов. Третья строка должна содержать L целых чисел — номера городов, в которые следует направить магов.

Заметьте, что вам *не требуется* минимизировать L. Если решений несколько, выведите любое.

Пример

kingdom.in	kingdom.out
8 3 0 1 1 1 1 2 2 2 1 0 1 1 2 1 2 2 1 1 0 1 2 2 1 2 1 1 1 0 2 2 2 1 1 2 2 2 0 2 1 1 2 1 2 2 2 0 2 1 2 2 1 2 1 2 0 2 2 2 2 1 1 1 2 0	1 3 1 3 8

Задача D. Свет

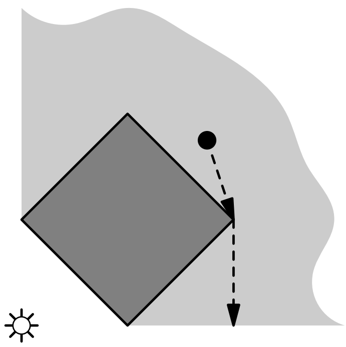
Имя входного файла:	`light.in`
Имя выходного файла:	`light.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

В точке $(0,0)$ координатной плоскости расположена лампочка, которая представляет собой точечный источник света. Неподалеку от лампочки находится дом Пети, который представляет собой выпуклый многоугольник с N вершинами. Сам Петя находится в точке с координатами (x,y).

Петя хочет увидеть свет. Для этого ему требуется оказаться в такое точке, что отрезок, соединяющей ее с началом координат, не пересекается с домом Пети (но может его касаться, в частности, проходить вдоль стороны многоугольника дома).

Петя может перемещаться по плоскости со скоростью v. Разумеется, Петя не может проходить сквозь дом (хотя он может перемещаться по его границе).

Выясните, какое минимальное время требуется Пете, чтобы оказаться в освещенной точке.



Формат входных данных

Первая строка входного файла содержит координаты Пети — два неотрицательных вещественных числа, не превышающих 1000, и его скорость v — вещественное число, $10^{-2} \le v \le 10^4$.

Вторая строка входного файла содержит N — число вершин в многоугольнике, задающем Петин дом ($3 \le N \le 100$). Следующие N строк содержат по два вещественных числа и задают координаты вершин многоугольника в порядке обхода их против часовой стрелки. Все координаты неотрицательны и не превышают 1000.

Гарантируется, что входные данные корректны, в частности многоугольник выпуклый, и никакие три его последовательные вершины не лежат на одной прямой. Также гарантируется, что и Петя и лампочка находятся снаружи от многоугольника, в частности, не находятся на его границе. Расстояние от точки, где находится Петя, до многоугольника и от начала координат до многоугольника не меньше 10^{-2}, расстояние от Пети до начала координат не меньше 10^{-2}.

Формат выходных данных

Выведите в выходной файл минимальное время, за которое Петя сможет попасть в освещенную точку. Ваш ответ должен отличаться от правильного не более чем на 10^{-4}.

Пример

light.in	light.out
3.5 3.5 1.0 4 2.0 0.0 4.0 2.0 2.0 4.0 0.0 2.0	3.58113883008418967

Задача E. Упаковка

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

В одну транспортную компанию поступил заказ на перевозку двух ящиков из одного города в другой. Для перевозки ящики решено было упаковать в специальный контейнер.

Ящики и контейнер имеют вид прямоугольных параллелепипедов. Длина, ширина и высота первого ящика — l_1, w_1 и h_1, соответствующие размеры второго ящика — l_2, w_2 и h_2. Контейнер имеет длину, ширину и высоту l_c, w_c и h_c.

Поскольку ящики содержат хрупкое оборудование, после упаковки в контейнер каждый из них должен остаться в строго вертикальном положении. Таким образом, ящики можно разместить рядом или один на другом. Для надежного закрепления в контейнере стороны ящиков должны быть параллельны его сторонам. Иначе говоря, если исходно ящики были расположены так, что все их стороны параллельны соответствующим сторонам контейнера, то каждый из них разрешается перемещать и поворачивать относительно вертикальной оси на угол, кратный 90 градусам.

Разумеется, после упаковки оба ящика должны полностью находиться внутри контейнера и не должны пересекаться.

Выясните, можно ли поместить ящики в контейнер, с выполнением указанных условий.

Формат входных данных

Первая строка входного файла содержит l_1, w_1 и h_1, вторая — l_2, w_2 и h_2, третья — l_c, w_c и h_c. Все размеры — целые положительные числа, не превышающие 1000. Числа в строках разделены пробелами.

Формат выходных данных

Выведите `YES`, если ящики можно упаковать в контейнер и `NO` в противном случае.

Пример

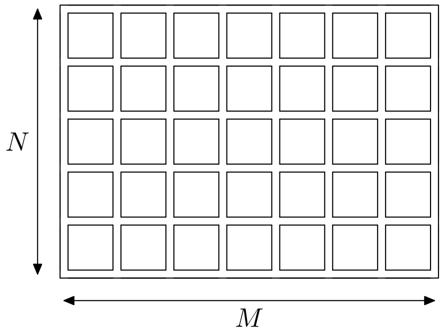
packing.in	packing.out
2 2 3 3 3 3 3 5 3	YES
2 3 3 3 2 3 4 4 4	YES
4 1 2 3 3 2 4 3 4	YES
1 1 4 1 1 3 10 10 3	NO
3 2 2 3 1 2 5 2 3	NO

В первых двух примерах ящики можно разместить рядом, при этом во втором один из них следует повернуть на 90 градусов. В третьем примере ящики можно разместить один на другом. В четвертом примере первый ящик слишком высокий и не влезает в контейнер. В пятом примере ящики нельзя разместить ни рядом, ни один на другом.

Задача F. Манхаттанский полицейский

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

Недавно Билл устроился на работу полицейским. Теперь ему предстоит каждый вечер обходить свой участок, который представляет собой прямоугольник, состоящий из $N \times M$ кварталов. Каждый квартал имеет вид квадрата размером 100×100 метров, кварталы отделены друг от друга прямыми улицами.



Таким образом, через участок Билла проходит $N + 1$ улица, идущая с запада на восток и $M + 1$ улица, идущая с севера на юг. Перекрестки разбивают улицы на $(N + 1)M + (M + 1)N$ отрезков, каждый из которых имеет длину 100 метров.

Совершая обход, Билл выходит из полицейского управления, расположенного около юго-западного угла его участка, обходит свой участок и возвращается в управление. Во время обхода Билл должен пройти по каждому отрезку улицы на территории своего участка как минимум один раз. Известно, что во время обхода Билл проходит отрезок длиной 100 метров за одну минуту. Выясните, какое минимальное число минут потребуется Биллу, чтобы совершить обход.

Формат входных данных

Входной файл содержит целые числа N и M, разделенные пробелом ($1 \le N, M \le 10\,000$).

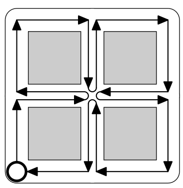
Формат выходных данных

В выходной файл выведите минимальное время, за которое Билл может совершить обход.

Пример

police.in	police.out
1 1	4
2 2	16
4 3	38

Один из возможных оптимальных путей для Билла во втором примере показан на рисунке.



Задача G. Электронная таблица

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

Электронная таблица представляет собой прямоугольную таблицу, левая и верхняя граница которой зафиксированы, а правая и нижняя отсутствуют, таким образом она бесконечна вправо и вниз. В каждой ячейке таблицы может быть записано какое-либо значение. Значение ячейки — это произвольная последовательность символов с кодами от 32 до 126.

При сохранении таблицы в файл, она записывается в специальном формате. Ячейки перечисляются в файле начиная с левой верхней по строкам сверху вниз, внутри строки слева направо. В каждой строке перечисляется несколько подряд идущих ячеек начиная с первой, при этом в частности перечисляются все непустые ячейки данной строки. Всего в файл выводится несколько подряд идущих строк, начиная с первой, при этом выводятся как минимум все строки, в которых содержится хотя бы одна непустая ячейка.

В файле в указанном порядке записываются значения ячеек. Значения соседних ячеек разделяются между собой символом ‘,’ (запятая), строки таблицы отделяются друг от друга символом ‘;’ (точка с запятой). После последней клетки идет символ ‘.’ (точка). После каждого из этих символов может немедленно следовать один перевод строки, который должен игнорироваться. Другие переводы строки во входном файле не встречаются.

Если в значении ячейки встречается один из символов ‘,’, ‘;’, ‘.’ или ‘\’, то при записи в файл записывается два символа — сначала символ ‘\’, а затем данный символ. Соответственно, запятая, точка с запятой и точка, которые идут непосредственно после ‘\’, не являются разделителями значений ячеек. В частности после них не может следовать перевода строки.

Каждая ячейка относится к одному из трех типов: числовая, строковая, пустая. Пустая ячейка — это ячейка, значение которой является пустой строкой. Числовая ячейка содержит целое число из диапазона от -32768 до 32767 включительно. Число должно быть записано без ведущих нулей и лишних знаков ‘+’ или ‘−’ (знак ‘−’ должен быть только у отрицательных чисел, причем ровно один). Любая другая ячейка относится к строковому типу. Так, например, к строковому типу относятся ячейки, содержащие следующие значения: 01 (содержит ведущий нуль), 55000 (не входит в указанный диапазон), а также ячейка, содержащая один символ ‘пробел’.

Столбец таблицы называется пустым, если все ячейки, которые он содержит — пустые. Столбец таблицы называется числовым, если он содержит только числовые и пустые ячейки. В противном случае столбец называется строковым. Требуется для каждого столбца таблицы, начиная с первого и до последнего непустого, определить, к какому типу он относится.

Формат входных данных

Входной файл содержит электронную таблицу, его размер не превышает 32767 байт.

Формат выходных данных

В выходной файл для всех столбцов, начиная с первого и до последнего непустого столбца, выведите их тип, разделив значения запятыми, и в конце поставьте точку. В качестве типа столбца выведите одно из следующих значений: “`EMPTY`”, если столбец является пустым, “`NUMBER`”, если столбец является числовым, “`STRING`”, если столбец является строковым.

Пример

table.in	table.out
`;,12;,, ;` `;,17,2,,-1\.0.`	`EMPTY,NUMBER,STRING,EMPTY,STRING.`
`.`	`.`

Таблица для первого примера приведена ниже. Символ 'пробел' обозначен как $\square$.

					$\ldots$
	12				$\ldots$
		$\square$			$\ldots$
					$\ldots$
	17	2		-1.0	$\ldots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Задача H. Терминал

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

Специальный терминал, разработанный в лаборатории, где работает Дима, представляет собой горизонтальный прямоугольник, состоящий из $m \times n$ ячеек, каждая из которых может содержать произвольное целое число. Ячейки занумерованы парами чисел, левая верхняя ячейка имеет номер $(1, 1)$, правая нижняя — (m, n).

Специальное устройство ввода, сконструированное специально для этого терминала, позволяет отправлять терминалу две команды: $A(r, c, d, v)$ и $B(r_1, c_1, r_2, c_2, d, v)$.

Рассмотрим сначала команду A. Параметры r и c изменяются в пределах от 1 до m и от 1 до n соответственно и указывают, к какой ячейке применяется команда. Параметр d может принимать значение из множества $\{L, R, U, D\}$ и задает направление, в котором применяется команда: влево, вправо, вверх или вниз соответственно. Параметр v представляет собой целое неотрицательное число. В результате выполнения команды значения во всех ячейках, находящихся в направлении d от ячейки (r, c), включая эту ячейку, увеличиваются на v.

Например, если терминал имеет размер 5×4, то после выполнения команды $A(3, 2, R, 3)$ значения в ячейках $(3, 2)$, $(3, 3)$ и $(3, 4)$ увеличатся на 3, а после команды $A(2, 1, U, 2)$ значения в ячейках $(2, 1)$ и $(1, 1)$ увеличатся на 2.

Рассмотрим теперь команду B. Первые четыре ее параметра являются целыми числами и удовлетворяют условиям $1 \le r_1 \le r_2 \le m$ и $1 \le c_1 \le c_2 \le n$. Параметры d и v могут принимать те же значения, что и соответствующие параметры команды A. Команда B выполняется следующим образом: для всех пар (r, c), таких, что $r_1 \le r \le r_2$ и $c_1 \le c \le c_2$ выполняется команда $A(r, c, d, v)$.

Исходно все ячейки терминала содержат нули. Выведите содержимое терминала после выполнения заданной последовательности команд.

Формат входных данных

Первая строка входного файла содержит числа m и n, $(1 \le m, n \le 200)$. Следующая строка содержит числа k — количество команд, которые следует обработать $(0 \le k \le 40\,000)$. Следующие k строк содержат описания команд. Первый символ каждой строки задает тип команды, затем следует пробелы и параметры команды, каждые два последовательных параметра разделены ровно одним пробелом. Параметр v каждой команды неотрицателен и не превышает 100.

Общее число команд A, которое потребуется выполнить на терминале, включая команды, которые придется выполнить при выполнении команд B, не превышает $5 \cdot 10^6$.

Формат выходных данных

Выведите в выходной файл m строк, по n чисел в каждой — содержимое терминала после выполнения указанной последовательности команд.

Пример

<table>
<tr><th>terminal.in</th><th>terminal.out</th></tr>
<tr><td><pre>5 4
4
A 2 2 D 1
A 3 4 L 2
B 2 3 3 4 U 13
B 1 1 2 1 R 5</pre></td><td><pre>5 5 31 31
5 6 31 31
2 3 15 15
0 1 0 0
0 1 0 0</pre></td></tr>
</table>

Задача I. Рейсы во времени

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

Между N населенными пунктами совершаются пассажирские рейсы на машинах времени.

В момент времени 0 вы находитесь в пункте A. Вам дано расписание рейсов. Требуется оказаться в пункте B как можно раньше (то есть в наименьший возможный момент времени).

При этом разрешается делать пересадки с одного рейса на другой. Если вы прибываете в некоторый пункт в момент времени T, то вы можете уехать из него любым рейсом, который отправляется из этого пункта в момент времени T или позднее (но не раньше).

Формат входных данных

Первая строка входного файла содержит число N — количество населенных пунктов ($1 \le N \le 1000$). Вторая строка содержит два числа A и B — номера начального и конечного пунктов. Третья строка содержит число K — количество рейсов ($0 \le K \le 1000$). Следующие K строк содержит описания рейсов, по одному на строке. Каждое описание представляет собой четверку целых чисел. Первое число каждой четверки задает номер пункта отправления, второе — время отправления, третье — пункт назначения, четвертое — время прибытия. Номера пунктов — натуральные числа из диапазона от 1 до N. Пункт назначения и пункт отправления могут совпадать. Время измеряется в некоторых абсолютных единицах и задается целым числом, по модулю не превышающим 10^9. Поскольку рейсы совершаются на машинах времени, то время прибытия может быть как больше времени отправления, так и меньше, или равным ему.

Гарантируется, что входные данные таковы, что добраться из пункта A в пункт B всегда можно.

Формат выходных данных

Выведите в выходной файл минимальное время, когда вы сможете оказаться в пункте B.

Пример

<table>
<tr><th>time.in</th><th>time.out</th></tr>
<tr><td><pre>2
1 1
2
1 1 2 10
1 10 1 9</pre></td><td><pre>0</pre></td></tr>
<tr><td><pre>1
1 1
3
1 3 1 -5
1 -5 1 -3
1 -4 1 -10</pre></td><td><pre>-10</pre></td></tr>
<tr><td><pre>5
1 2
6
1 0 3 10
4 2 2 -10
4 14 2 -7
3 10 2 10
2 0 4 2
3 10 4 12</pre></td><td><pre>-10</pre></td></tr>
</table>