,

Задача A. Простая задача

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Найдите оптимальное решение задачи линейного программирования.

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \\ x_1 \geqslant 0 \\ \ldots \\ x_n \geqslant 0 \\ c_1x_1 + \ldots + c_nx_n \to \max \end{cases}$$

Формат входных данных

Первая строка входного файла содержит два целых числа: n и m — количество переменных и количество уравнений ($1 \leqslant n, m \leqslant 5$). Следующие m строк содержат по $n + 1$ целому числу: $a_{i1}, \ldots, a_{in}, b_i$. Следующая строка содержит n целых чисел: $c_1, \ldots, c_n$. Все числа во входном файле не превышают 100 по модулю.

Формат выходных данных

Выведите одно число: максимальное значение $c_1x_1 + \ldots + c_nx_n$.

Гарантируется, что решение существует и максимальное значение достигается.

Примеры

стандартный ввод	стандартный вывод
2 2 1 2 3 2 1 3 1 1	2.000000000
1 1 1 2 -2	-0.000000000
1 1 4 5 3	3.750000000

,

Задача B. Простая задача 2

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Найдите оптимальное решение задачи линейного программирования.

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \\ x_1 \geqslant 0 \\ \ldots \\ x_n \geqslant 0 \\ c_1x_1 + \ldots + c_nx_n \to \max \end{cases}$$

Формат входных данных

Первая строка входного файла содержит два целых числа: n и m — количество переменных и количество уравнений ($1 \leqslant n, m \leqslant 5$). Следующие m строк содержат по $n + 1$ целому числу: $a_{i1}, \ldots, a_{in}, b_i$. Следующая строка содержит n целых чисел: $c_1, \ldots, c_n$. Все числа во входном файле не превышают 100 по модулю.

Формат выходных данных

Выведите одно число: максимальное значение $c_1x_1 + \ldots + c_nx_n$. Если решения нет, выведите "`No solution`". Если можно получить сколь угодно большое значение, выведите "`Unbounded`".

Примеры

стандартный ввод	стандартный вывод
2 2 1 2 3 2 1 3 1 1	2.000000000000000
2 1 -1 -1 0 1 1	Unbounded
2 1 1 1 -1 1 1	No solution

,

Задача C. Простая задача с хорошими тестами

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Найдите оптимальное решение задачи линейного программирования.

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \\ x_1 \geqslant 0 \\ \ldots \\ x_n \geqslant 0 \\ c_1x_1 + \ldots + c_nx_n \to \max \end{cases}$$

Формат входных данных

Входной файл содержит один или несколько тестов. На первой строке число тестов, далее сами тесты. Каждый тест описывается следующим образом. Первая строка теста содержит два целых числа: n и m — количество переменных и количество неравенств ($1 \leqslant n, m \leqslant 5$). Следующие m строк содержат по $n + 1$ целому числу: $a_{i1}, \ldots, a_{in}, b_i$. Следующая строка содержит n целых чисел: $c_1, \ldots, c_n$. Все числа во входном файле целые и не превышают 100 по модулю.

Формат выходных данных

Для каждого теста выведите одну строку.

Выведите максимальное значение $c_1x_1 + \ldots + c_nx_n$, пробел, двоеточие, пробел, сами числа $x_1x_2 \ldots x_n$. Все числа выводите с максимальной точностью. Если решения нет, выведите "`No solution`". Если можно получить сколь угодно большое значение, выведите "`Unbounded`".

Пример

стандартный ввод	стандартный вывод
4 2 2 1 2 3 2 1 3 1 1 2 1 -1 -1 0 1 1 2 1 1 1 -1 1 1 2 2 1 2 3 2 1 3 -10 1	2.000000000 : 1.000000000 1.000000000 Unbounded No solution 1.500000000 : 0.000000000 1.500000000

,

Задача D. Двойственная задача линейного программирования

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

В этой задаче вам нужно построить задачу, двойственную к задаче линейного программирования, заданной в общем виде.

Задача линейного программирования задаётся следующим образом. Для $i = 1, 2, \ldots, n$ пусть x_i — переменная. Некоторые x_i могут быть обязаны быть неотрицательными или неположительными. Требуется минимизировать или максимизировать сумму $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ при заданном наборе ограничений.

Пусть $A = (a_{ij})$ — матрица размера $m \times n$. Обозначим произведение $a_{i,1}x_1 + a_{i,2}x_2 + \ldots + a_{i,n}x_n$ как A_ix. Каждое ограничение имеет вид $A_ix \geqslant b_i$, $A_ix \leqslant b_i$ или $A_ix = b_i$.

Двойственная к этой задача линейного программирования определяется как задача с переменными y_i для $i = 1, 2, \ldots, m$ (число двойственных переменных равно числу ограничений исходной задачи). Если исходная задача была задачей минимизации, то двойственная является задачей максимизации, и наоборот. Целевая функция двойственной задачи — это $b_1y_1 + b_2y_2 + \ldots + b_my_m$.

Каждая двойственная переменная связана с ограничением исходной задачи. Если исходная задача была задачей максимизации, то переменные, связанные с ограничениями $A_ix \leqslant b_i$, должны быть неотрицательными, а переменные, связанные с ограничениями $A_ix \geqslant b_i$, должны быть неположительными. Если же исходная задача была задачей минимизации, то переменные, связанные с ограничениями $A_ix \geqslant b_i$, должны быть неотрицательными, а связанные с $A_ix \leqslant b_i$ — неположительными. Переменные, связанные с ограничениями $A_ix = b_i$, могут быть произвольными в обоих случаях.

Ограничения двойственной задачи используют матрицу A^T (A транспонированную) вместо A. Ограничения имеют один из видов $A_i^Ty \leqslant c_i$, $A_i^Ty \geqslant c_i$ или $A_i^Ty = c_i$. Тип каждого ограничения нужно определить исходя из того, что задача, двойственная к двойственной, — это снова исходная задача.

По заданной исходной задаче линейного программирования выведите двойственную к ней.

Формат входных данных

В первой строке заданы n и m ($1 \leqslant n, m \leqslant 100$).

Во второй строке задана целевая функция исходной задачи. Первое слово строки — либо `min`, либо `max`, далее следует целевая функция. Все c_i целые, x_i записывается как `xi`, знак умножения опускается, знак минус может заменять знак плюс, когда нужно указать, что c_i отрицательно. Слагаемые с $c_i = 0$ опускаются. Если все c_i равны нулю, в качестве целевой функции задаётся `0`. Слагаемые с $c_i = \pm 1$ записываются без символа `1`.

Третья строка содержит слово `with`.

Следующие n строк содержат условия неотрицательности и неположительности переменных. Если переменная может быть произвольной, используется слово `arbitary`.

Следующая строка содержит слово `under`.

Следующие m строк содержат ограничения. Все a_{ij} и b_i целые, знак умножения опускается, слагаемые с $a_{ij} = 0$ опускаются, и если для некоторого i все a_{ij} равны нулю, в качестве левой части равенства используется `0`. Слагаемые с $a_{ij} = \pm 1$ записываются без символа `1`.

Во входных данных нет лишних пробелов.

Формат выходных данных

Выведите задачу, двойственную к заданной во входных данных, в том же формате. Помните, что первая строка должна содержать число переменных и ограничений двойственной задачи, так чтобы полученный вывод являлся корректным входным файлом для этой задачи, за исключением того, что вместо `x` используется `y`.

Переменные во всех выражениях нужно перечислять в порядке возрастания их номеров и опускать слагаемые, равные нулю. Коэффициент ± 1 нужно опускать там, где это требуется, оставляя только знак. Нельзя печатать +, если следующее слагаемое имеет отрицательный коэффициент.

Пример

стандартный ввод	стандартный вывод
3 4 min 2x1-x3 with x1>=0 x2 arbitary x3<=0 under x1+x2+x3>=0 3x2=3 -x1-x2+100x3<=-4 0=0	4 3 max 3y2-4y3 with y1>=0 y2 arbitary y3<=0 y4 arbitary under y1-y3<=2 y1+3y2-y3=0 y1+100y3>=-1

Замечание

В примере исходная задача — задача минимизации с $n = 3$ переменными и $m = 4$ ограничениями, поэтому двойственная — задача максимизации с 4 переменными и 3 ограничениями. Ограничение 1 ($\geqslant$) даёт $y_1 \geqslant 0$, ограничение 3 ($\leqslant$) даёт $y_3 \leqslant 0$, а ограничения-равенства 2 и 4 дают свободные переменные y_2 и y_4. Каждое двойственное ограничение получается из столбца матрицы A в сравнении с соответствующим c_j: переменная $x_1 \geqslant 0$ даёт ограничение $\leqslant$, $x_3 \leqslant 0$ даёт ограничение $\geqslant$, а свободная переменная x_2 даёт равенство.

,

Задача E. Road times

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Дорожная сеть страны – ориентированный граф. У каждого ребра есть длина, целое число от 1 до 1000, и ограничение на максимальную скорость, вещественное число от 30 до 60, в километрах в час. Длины вам известны, а ограничения – нет. Известно, что когда водитель такси берётся доставить пассажира из вершины a в вершину b, он осуществляет перевозку строго по кратчайшему пути между вершинами и едет с максимальной допустимой скоростью. Длина пути – сумма длин рёбер. Также известно, что $\forall a, b\ \exists$ *единственный* кратчайший путь из a в b.

Ваша задача – по длинам рёбер и уже сделанным поездкам $a_i\ b_i\ time_i$ оценить минимальное и максимальное время в пути между вершинами $c_j\ d_j$

Формат входных данных

На первой строке число вершин n ($1 \leqslant n \leqslant 30$).
Вершины нумеруются числами от 0 до $n - 1$.
Следующие n строк содержат матрицу $n \times n$ длин дорог. Дороги односторонние.
Отсутствие дорог обозначено числом -1, длины дорог целые от 1 до 1 000.
Всего не более 100 дорог.
Далее идёт число r ($1 \leqslant r \leqslant 100$) и r уже известных маршрутов $a_i\ b_i\ time_i$, $time_i \in \mathbb{R}$.
Времена в минутах (не в часах).
Затем число запросов q ($1 \leqslant q \leqslant 100$) и q строк $c_j\ d_j$.

Формат выходных данных

Для каждого запроса выведите четыре числа – откуда, куда, min время, max время.

Пример

стандартный ввод	стандартный вывод
3 0 50 -1 55 0 40 -1 40 0 1 0 2 120 3 0 1 1 2 1 0	0 1 50.000000000000000 80.000000000000000 1 2 40.000000000000000 69.999999999999986 1 0 55.000000000000000 110.000000000000000

,

Задача F. Гиперраздление

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
Ограничение по памяти:	256 мегабайт

Вам даны N d-мерных точек. Нужно разделить их $d-1$ мерной плоскостью, проходящей через точку $(0,\ldots,0)$. Гарантируется, что решение всегда существует.

Формат входных данных

В первой строке N ($1 \leqslant N \leqslant 5\,000$), d ($1 \leqslant d \leqslant 10$). Далее N строк содержат по $d+1$ чисел. Первые d чисел — вещественные числа, координаты очередной точки $x_{j,1}, x_{j,2}, \ldots, x_{j,d}$. Последнее число — целое число y_j ($|y_j| = 1$). Число y_j определяет положение точки относительно плоскости, читайте дальше для более точного понимания.

Координаты точек по модулю не превосходят 100.0. Все вещественные числа содержат не более 6 знаков после десятичной точки.

Формат выходных данных

Выведите d вещественных чисел $n_1, n_2, \ldots, n_d$ — нормаль плоскости. Нормаль — вектор произвольной **положительной** длины. Для каждой точки $sign(\sum_{i=1}^{d} x_{j,i} \cdot n_i)$ должен быть равен ее y_j. Если решений несколько, можно вывести любое.

Ответ будет считаться корректным, если для каждой точки $|\sum_{i=1}^{d} x_{j,i} \cdot n_i| \geqslant 10^{-4}|n|$, где $|n|$ — длина вектора нормали.

Примеры

стандартный ввод	стандартный вывод
2 1 1 -1 -1 1	-1.000000000
4 3 -99.749 12.71 -61.33 1 61.7 17.00 -4.0 -1 -29.94 79.192 64.56 1 49.320 -65.178 71.788 -1	-0.891651465 0.415648308 -0.179427280

,

Задача G. Триатлон

Имя входного файла:	`tri.in`
Имя выходного файла:	`tri.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Триатлон — атлетическое соревнование, состоящее из трех последовательных этапов, которые суммарно должны быть пройдены как можно быстрее. Первый этап — плавание, второй — велогонка, а третий — бег.

Известна скорость каждого участника для каждого этапа. Жюри соревнования может выбрать длину каждого этапа любым образом так, чтобы каждый этап не был нулевой длины. В результате, иногда жюри может выбрать длины этапов так, чтобы определенный участник стал победителем соревнования.

Формат входных данных

В первой строке входного файла задано целое число N — количество участников соревнования ($1 \leqslant N \leqslant 100$). Следующие N строк содержат по 3 целых числа V, U и W ($1 \leqslant U, V, W \leqslant 10000$), определяющие скорость очередного участника на этапах соревнования.

Формат выходных данных

Для каждого участника выведите одну строку, содержащую слово Yes в случае если жюри может подобрать длины этапов так, чтобы этот участник стал победителем (т.е. этот участник — единственный, кто первый дошел до финиша), или слово No в противном случае.

Примеры

tri.in	tri.out
9 10 2 6 10 7 3 5 6 7 3 2 7 6 2 6 3 5 7 8 4 6 10 4 2 1 8 7	Yes Yes Yes No No No Yes No Yes

,

Задача H. Fishingprince снова играет с массивом

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	6 секунд
Ограничение по памяти:	1024 мегабайта

Пусть дана последовательность неотрицательных целых чисел a длины n в 1-индексации. Также даны два целых числа x, y. В течение отрезка времени в t секунд (t может быть любым положительным вещественным числом) можно выполнить одну из следующих операций:

- Выбрать $1 \leqslant i < n$, уменьшить a_i на $x \cdot t$ и уменьшить a_{i+1} на $y \cdot t$.
- Выбрать $1 \leqslant i < n$, уменьшить a_i на $y \cdot t$ и уменьшить a_{i+1} на $x \cdot t$.

Определим $f(a)$ как минимальное количество времени (может быть вещественным числом), необходимое для того, чтобы сделать все элементы последовательности неположительными (меньше или равными 0).

Например, если $x = 1$, $y = 2$, то массив $[3, 1, 1, 3]$ можно обработать за 3 секунды. Для этого нужно:

- В течение первых 1.5 секунд выполнять вторую операцию с параметром $i = 1$.
- В течение оставшихся 1.5 секунд выполнять первую операцию с параметром $i = 3$.

Можно доказать, что в этом случае невозможно сделать все элементы меньше или равными 0 меньше чем за 3 секунды, поэтому $f([3, 1, 1, 3]) = 3$.

Вам дана последовательность положительных целых чисел b длины n в 1-индексации. Также даны положительные целые числа x, y. Обработайте q запросов следующих двух типов:

- `1 k v`: изменить значение b_k на v.
- `2 l r`: вывести $f([b_l, b_{l+1}, \ldots, b_r])$.

Формат входных данных

Первая строка содержит два целых числа n и q ($2 \leqslant n \leqslant 2 \cdot 10^5$, $1 \leqslant q \leqslant 2 \cdot 10^5$).

Вторая строка содержит два целых числа x и y ($1 \leqslant x, y \leqslant 10^6$).

Третья строка содержит n целых чисел $b_1, b_2, \ldots, b_n$ ($1 \leqslant b_i \leqslant 10^6$).

Далее следует q строк. Каждая из этих q строк содержит по три целых числа. Первое число op равно 1 или 2.

Если оно равно 1, далее следуют два целых числа k и v ($1 \leqslant k \leqslant n$, $1 \leqslant v \leqslant 10^6$). Такой запрос означает, что нужно изменить значение b_k на v.

Если оно равно 2, далее следуют два целых числа l и r ($1 \leqslant l < r \leqslant n$). Такой запрос означает, что нужно вывести значение $f([b_l, b_{l+1}, \ldots, b_r])$.

Формат выходных данных

Для каждого запроса типа 2 выведите одно вещественное число — ответ на этот запрос. Ваш ответ будет считаться правильным, если его абсолютная или относительная погрешность не превосходит 10^{-9}.

Пример

стандартный ввод	стандартный вывод
4 3 1 2 3 1 1 4 2 1 4 1 1 1 2 1 3	3.500000000000000 1.000000000000000

,

Замечание

Разберём тестовый пример.

В первом запросе требуется вычислить $f([3,1,1,4])$. Ответ равен 3.5. Одной из оптимальных последовательностей операций является следующая:

- В течение первых 1.5 секунд выполнять вторую операцию с параметром $i = 1$.
- В течение оставшихся 2 секунд выполнять первую операцию с параметром $i = 3$.

В третьем запросе требуется вычислить $f([1,1,1])$. Ответ равен 1. Одной из оптимальных последовательностей операций является следующая:

- В течение первых 0.5 секунд выполнять вторую операцию с параметром $i = 1$.
- В течение оставшихся 0.5 секунд выполнять первую операцию с параметром $i = 2$.