Задача A. Ахроматическое число графа

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

Хроматическим числом графа называют минимальное количество цветов, которое необходимо для того, чтобы раскрасить его вершины таким образом, чтобы никакие две вершины одного цвета не были соединены ребром. Такая раскраска вершин графа называется *правильной*. Известно, что нахождение хроматического числа графа является трудной задачей.

Правильная раскраска графа в свою очередь называется *ахроматической*, если каждая пара различных цветов встречается на концах некоторого ребра. *Ахроматическим числом* графа называют максимальное количество цветов, которое можно использовать для раскраски вершин графа, чтобы полученная раскраска была ахроматической.

Например, ахроматическое число треугольника равно трем, поскольку если раскрасить все его вершины в различные цвета, каждая пара цветов будет встречаться на концах некоторого ребра.

Вам дано число n. Найдите ахроматическое число цикла длиной n и выведите соответствующую ахроматическую раскраску.

Формат входного файла

Входной файл содержит одно целое число n ($3 \le n \le 1000$).

Формат выходного файла

На первой строке выходного файла выведите одно число a — ахроматическое число цикла длиной n. Вторая строка должна содержать n целых чисел в диапазоне от 1 до a и описывать соответствующую правильную ахроматическую раскраску.5

Примеры

achromatic.in	achromatic.out
3	3 1 2 3

Задача B. Двоичный поиск

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

Двоичный поиск — это алгоритм, который используется для поиска заданного элемента в отсортированном массиве. Рассмотрим следующий псевдокод двоичного поиска («/» означает деление нацело):

```
вход: a[0..n - 1], x

l = 0;
r = n;
while (l < r - 1) {
    m = (l + r) / 2;
    if (a[m] <= x)
        l = m;
    else
        r = m;
}
if (a[l] == x)
    return true;
else
    return false;
```

Иногда оказывается, что двоичный поиск находит вхождение некоторого элемента в массив даже если массив не отсортирован. Вам задано число n. Требуется найти количество пар $\langle a, x\rangle$, где a представляет собой массив длины n, содержащий целые числа от 1 до n, а x — целое число от 1 до n, таких что приведенная процедура возвращает “`true`”, если ее запустить на массиве a и числе x в качестве аргументов.

Формат входного файла

Входной файл содержит одно целое число n ($1 \le n \le 1000$).

Формат выходного файла

Выведите одно целое число — ответ на поставленную задачу.

Примеры

binary.in	binary.out
1	1
2	5

Задача C. Игра с монетками

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

Аня и Бетти играют в следующую игру с монетками. Исходно они выкладывают $m \times n$ монеток на столе в виде прямоугольной сетки размером $m \times n$, некоторые монеты оказываются вверх орлом, некоторые — вверх решкой. Будем индексировать монеты сетки парами чисел (x, y), где $1 \le x \le m$ и $1 \le y \le n$. После этого игроки совершают ходы по очереди, Аня ходит первой.

Ход игрока состоит в следующем: игрок выбирает монету, которая лежит вверх решкой, и переворачивает ее. Пусть была перевернута монета в позиции (i, j) ($1 \le i \le m$, $1 \le j \le n$). После этого игрок выбирает также числа i_1 и j_1, для которых выполнены неравенства $0 \le i_1 < i$, $0 \le j_1 < j$ и переворачивает монеты в позициях (i_1, j), (i, j_1) и (i_1, j_1) (если какие-либо из этих монет отсутствуют, поскольку одна из координат равна нулю, то эти позиции игнорируются).

Игрок, который не может сделать ход, поскольку все монеты лежат вверх орлом, проигрывает.

По заданному исходному расположению монет выясните, кто выиграет при оптимальной игре обоих игроков, и если это Аня, то какой первый ход она должна сделать.

Формат входного файла

Первая строка входного файла содержит m и n ($1 \le m, n \le 50$). Следующие m строк содержат по n символов, j-й из них в i-й строке равен «`1`», если монета в позиции (i, j) лежит решкой вверх и «`0`», если она лежит орлом вверх.

Формат выходного файла

Первая строка выходного файла должна содержать имя победителя игры («`Ann`» или «`Betty`»). Если побеждает Аня, то на второй строке выведите i и j, а на т ретьей строке i_1 и j_1. Эти числа должны описывать первый ход, который должна сделать Аня, чтобы выиграть. Если возможных выигрышных ходов несколько, выведите любой.

Примеры

coins.in	coins.out
3 3 000 000 001	Ann 3 3 0 0
2 2 11 01	Betty

Задача D. Резать!

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

Держа в руках листок бумаги в клетку, у некоторых из нас возникает желание разрезать его. В данной задаче вам предлагается поддаться этому желанию!

К вам у нас есть лишь несколько просьб. Во-первых, разрезать листок можно только по линиям сетки. Во-вторых, каждый из получившихся кусочков бумаги должен состоять ровно из k клеток исходного листа.

Формат входного файла

Первая строка входного файла содержит два целых числа n и m ($1 \le n, m \le 100$) — соответственно, высоту и ширину листа бумаги в клетках. Вторая строка входного файла содержит число k ($1 \le k \le 1000$).

Формат выходного файла

Если данный листок бумаги нельзя разрезать указанным образом, в выходной файл выведите «-1».

Иначе, вам следует вывести n строк по m целых чисел в каждой — для каждой из клеток исходного листа бумаги выведите номер кусочка, в котором эта клетка оказалась. Кусочки следует нумеровать последовательными натуральными числами, начиная с единицы.

Примеры

cut.in	cut.out
2 2 1	1 2 4 3
2 2 2	1 1 2 2
2 2 3	-1
4 4 4	1 1 1 4 1 4 4 4 2 2 3 3 2 2 3 3

Задача E. Диаграмма

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

Руководство компании, в которой работает Вася, уделяет много внимания разнообразной статистике. Сбором информации и последующей ее обработкой занимается специальный отдел, в котором и работает Вася. Недавно ему поручили написать специальную программу по визуализации собранных данных.

После продолжительных диспутов, было решено использовать лепестковую диаграмму. Правила ее построения довольно просты. Предположим, что исследуемый объект имеет n характеристик. Про каждую из них известно p_i — то, на сколько процентов от максимального значения объект соответствует этой характеристике. Для того, чтобы отобразить эти данные на лепестковой диаграмме, проведем из начала координат n лучей так, чтобы углы между соседними были равными. На i-м луче отметим точку так, чтобы расстояние от нее до начала координат было равно p_i. Лучи нумеруются в порядке обхода против часовой стрелки.

Формат входного файла

Первая строка входного файла содержит одно целое число n — количество характеристик ($3 \le n \le 100$). Вторая строка выходного файла содержит n целых чисел $p_1, p_2, \ldots, p_n$ — степени соответствия в процентах ($1 \le p_i \le 100$).

Формат выходного файла

Выведите вершины лепестковой диаграммы в порядке обхода против часовой стрелки — n строк, каждая из которых содержит пару вещественных чисел, разделенных пробелом. Первая вершина должна лежать на оси Ox и иметь положительную абсциссу.

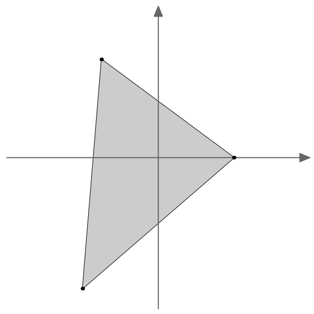
Ответ считается верным, если абсолютная погрешность не превосходит 10^{-6}.

Примеры

diagram.in	diagram.out
3 2 3 4	2.000000 0.000000 -1.500000 2.598076 -2.000000 -3.464102
3 20 30 40	20.000000 0.000000 -15.000000 25.980762 -20.000000 -34.641016
4 100 50 100 50	100 0 0 50 -100 0 0 -50

Примечание

Первый пример:



Задача F. Войны планет

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

Все большую и большую популярность набирает игра «Войны планет». Правила игры предельно просты. На плоскости расположены n планет, каждая из которых характеризуется координатами своего место положения x_i, y_i и скоростью производства космических кораблей s_i. В игру играют два игрока. Изначально каждый игрок имеет в своем распоряжении одну планету, а все остальные планеты в начале игры являются нейтральными. Также на каждой планете в начале находится некоторое число кораблей p_i. На нейтральных планетах — это нейтральные войска, а на планетах игроков — это войска соответствующих игроков.

Игра состоит из раундов. В начале каждого раунда на планетах, не являющихся нейтральными, появляются дополнительные s_i кораблей. Корабли мгновенно оказываются в распоряжении игрока, владеющего соответствующей планетой.

После этого игроки начинают пересылать войска с планеты на планету, при этом игрок может посылать корабли только с планет, которые ему принадлежат, и только не больше кораблей, чем на этой планете есть. Если игрок пересылает корабли с планеты с координатами (sx, sy) на планету с координатами (dx, dy), то корабли прибывают к планете назначения после $\lceil\sqrt{(sx - dx)^2 + (sy - dy)^2)}\rceil$ раундов ($\lceil a \rceil$ означает минимальное целое число, не меньшее a).

Затем происходит прибытие войск на планеты.

В конце раунда, если на какой-то планете оказываются войска обоих игроков, то они вступают в бой. В результате сражения у каждого из игроков уничтожается $x = \min(p_1, p_2)$ кораблей, где p_1 — число кораблей первого игрока на этой планете, а p_2 — второго. После этого, если на планете остались войска игрока и нейтральные войска, то в бой вступают они, при этом бой происходит по тем же правилам.

После всех сражений, если на планете более нуля кораблей какого-то игрока, то он захватывает планету. Иначе владелец планеты остается неизменным. На этом раунд заканчивается.

Напишите программу, которая обрабатывала бы события игры, а также могла отвечать на запросы про i-ую планету, сколько на ней войск и кто ею владеет.

Формат входного файла

В первой строке входного файла целое число n ($2 \le n \le 100$) — число планет. Далее следуют n строк, по четыре целых числа x, y, s и p в каждой. Числа x и y ($|x|, |y| \le 20$) — координаты соответствующей планеты, s ($0 \le s \le 10$) — скорость производства кораблей, p ($0 \le p \le 100$) — количество кораблей на планете в начале игры. В следующей строке следуют два целых числа — номера планет первого и второго игроков. Планеты нумеруются с единицы в порядке появления во входном файле. Гарантируется, что координаты всех планет различны.

В следующей строке входного файла одно целое число m ($0 \le m \le 1000$) — количество событий и запросов, которые необходимо обработать. Каждый запрос занимает отдельную строку и начитается и с целого числа t, характеризующего тип события/запроса.

Если $t = 0$, то это означает конец раунда. Считается что все сражения происходят именно в момент конца раунда.

Если $t = 1$, то происходит пересылка войск. Тогда в строке далее идут три целых положительных числа sp, dp и $ships$ — номер планеты, с которой отправляются корабли, номер планеты, на которую отправляются корабли, и число отправляемых кораблей соответственно. Гарантируется, что на планете sp будет не менее чем $ships$ кораблей, планета sp не будет нейтральной и планеты sp и dp не совпадают.

Если $t = 2$, то далее идет целое число $planet$ — номер планеты, о которой запрашивается информация.

Формат выходного файла

Для каждого запроса о планете выведите в отдельной строке ее владельца и количество войск на ней на момент запроса. Следуйте формату примера. Считается что все запросы происходят после производства кораблей и до прибытия войск на планеты.

Примеры

planetwars.in	planetwars.out
3 0 0 1 100 1 0 10 10 2 0 0 100 1 3 26 1 1 2 60 1 3 2 50 2 1 2 2 2 3 0 2 1 2 2 2 3 0 2 1 2 2 2 3 1 1 2 1 2 1 2 2 2 3 0 0 2 1 2 2 2 3 1 3 2 31 0 0 2 2	Player1 41 Neutral 10 Player2 50 Player1 42 Neutral 10 Player2 50 Player1 43 Neutral 0 Player2 50 Player1 42 Neutral 0 Player2 50 Player1 44 Player1 11 Player2 50 Player2 20

Примечание

В примере войска, вылетающие в первом раунде, долетают до планеты 2 только во конце второго раунда. При этом в конце второго дня на планете 2 окажутся 10 нейтральных кораблей, 60 кораблей первого игрока и 50 — второго. После битвы кораблей игроков останутся 10 нейтральных кораблей и 10 кораблей первого игрока. После из сражения на планете 2 не будет ни одного корабля и она останется нейтральной.

В третьем раунде первый игрок высылает один корабль на планету 2, который в конце четвертого раунда ее захватывает.

В пятом раунде второй игрок посылает 31 корабль на вторую планету, но когда они прибывают на планету 2 на ней уже есть 31 корабль первого игрока. Корабли полностью уничтожают друг друга, но планета остается под контролем первого игрока.

Задача G. Плейлист

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

Делая домашние задания, Вася любит слушать музыку из своего любимого плейлиста. Плейлист содержит N песен, каждая из которых длится некоторое целое количество секунд.

Раньше Вася слушал песни в одном порядке, но ему это быстро надоело, так как он знал какую песню услышит следующей. К счастью, у его плеера есть функция «перемешивание». Она изменяет порядок воспроизведения песен в плейлисте случайным образом, но при этом песня не может начать играть второй раз, пока не проигран весь плейлист. Теперь Вася всегда слушает музыку с включенным «перемешиванием».

Когда, в очередной раз, Вася собрался делать домашние задания и начал слушать музыку, его отвлекли, и он вернулся только через T секунд. Ему стало интересно, может ли в данный момент проигрываться его любимая песня, и он попросил у Вас помощи в решении этой задачи.

Песня считается проигрываемой в момент времени T, если она будет играть хотя бы в течение ближайшей секунды.

Формат входного файла

Первая строка входного файла содержит два целых числа N и T $(1 \le N \le 100,\ 0 \le T < \sum_{i=1}^{N} l_i)$ — количество песен в плейлисте и количество времени в секундах, через которое Вася вернулся.

Вторая строка содержит N целых чисел l_i $(1 \le l_i \le 500)$ — продолжительности песен в секундах. Васина любимая песня идет первой в этом списке.

Формат выходного файла

В выходной файл выведите «Yes», если через T секунд возможно проигрывание Васиной любимой песни и «No» в противоположном случае.

Примеры

playlist.in	playlist.out
3 10 3 5 7	No
4 5 2 3 4 5	Yes

Задача H. Дороги

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

Дорожная сеть города Нью-Флетсити устроена довольно просто. Все дороги являются отрезками единичной длины с концами в точках с целыми координатами. Этот факт — своего рода достопримечательность Нью-Флетсити.

Недавно пришедший к власти мэр считает, что он тратит слишком много времени на дорогу из дома в мэрию и обратно. Он решил построить несколько новых дорог так, чтобы этот путь был как можно короче. Естественно, новые дороги должны также являться единичными отрезками с концами в целых точках.

Вам, как главному инженеру Нью-Флетсити, поручено вычислить минимальное количество дорог, которое придется построить для осуществления плана мэра.

Формат входного файла

Первая строка входного файла содержит целое число n — количество дорог в Нью-Флетсити ($0 \le n \le 100$). Далее следуют n строк с четырьмя целыми числами, разделенными пробелами: x_i, y_i, x_j, y_j — координаты начала и конца соответствующей дороги ($0 \le x_i, y_i, x_j, y_j \le 100$). Последняя строка содержит два целых числа m_x и m_y — координаты дома мэра ($0 \le m_x, m_y \le 100$). Мэрия расположена в точке $(0, 0)$.

Все дороги расположены либо по горизонтали, либо по вертикали, а длина каждой из этих дорог равна единице. Движение по дорогам возможно в обе стороны.

Формат выходного файла

В выходной файл на первой строке выведите число M — количество новых дорог, которые нужно построить в Нью-Флетсити.

Примеры

roads.in	roads.out
1 0 0 1 0 1 1	1
5 0 0 1 0 1 0 1 1 1 1 0 1 0 1 0 2 0 2 1 2 1 2	1

Задача I. Стена

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

После завоевания Эджландии правительства Флатландии и союзников решили разделить ее столицу на зоны влияния. После длительных переговоров решено было построить стену для отделения территории Флатландии от территории союзников. Из-за стремления лидеров завоевателей к совершенству, решено было построить стену в форме окружности.

Каждый из завоевателей отметил несколько точек на карте, которые он был хотел иметь в своей зоне влияния. Стену следует построить таким образом, чтобы точки, выбранные Флатландией, оказались с одной стороны от стены, а точки, выбранные союзниками — с другой. Стена будет достаточно тонкой и может пройти с любой стороны от точки, лежащей непосредственно через нее.

Помогите завоевателям построить стену.

Формат входного файла

Первая строка входного файла содержит n и m — количество точек, которые интересуют Флатландию и союзников, соответственно ($2 \le n, m \le 120$). Следующие n строк содержат по два целых числа: координаты точек, отмеченных Флатландией. Затем следует m строк, которые описывают точки союзников. Координаты не превышают 10^4 по модулю, все точки различны.

Формат выходного файла

Если можно построить стену, выведите «`YES`» на первой строке входного файла. На второй строке выведите три вещественных числа: координаты центра и радиус окружности стены. Ответ должен быть выведен с точностью не менее 10^{-6}, но рекомендуется выводить как можно больше знаков после десятичной точки.

Если построить стену нельзя, выведите «`NO`» на первой строке выходного файла.

Примеры

wall.in	wall.out
2 2 0 0 0 1 1 0 1 1	YES 0 0.5 0.5
2 2 0 0 1 1 1 0 0 1	YES 0.5 0.5 0.70710678118654752
4 2 0 0 2 2 2 0 0 2 1 1 5 5	NO