Задача A. Военная Академия

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

Военная академия Флатландии недавно получила запрос от правительства на подготовку n секретных агентов для важнейшей операции против их давнего дружеского соседа Поппландии. Каждый агент получит специальную миссию, i-я миссия требует чтобы *квалификация* агента, который будет ее выполнять, была как минимум p_i.

Главный инструктор академии выбрал n студентов академии и назначил каждому из них одну из миссий — i-й из выбранных студентов будет выполнять миссию i. Теперь для подготовки агентов необходимо провести несколько тренировок.

С каждым студентом можно проводить личные тренировки, кроме того, можно проводить групповые тренировки со всеми студентами вместе. В начале квалификация каждого студента равна нулю. Каждый студент характеризуется своей *обучаемостью* q_i и *общительностью* s_i. После личных тренировок в течение m часов квалификация студента вырастает на mq_i. После групповых тренировок в течении m часов квалификация студента вырастает на ms_i. Тренировки могут продолжаться произвольное неотрицательное количество часов (не обязательно целое).

Тренировки проводятся инструкторами. За час личной тренировки инструктору требуется заплатить a рублей, а за час групповой тренировки — b рублей.

Помогите руководству академии спланировать тренировки так, чтобы выполнить заказ правительства и минимизировать сумму, потраченную на оплату работы инструкторов.

Формат входного файла

Первая строка входного файла содержит три целых числа: n $(1 \le n \le 100)$, a и b $(1 \le a, b \le 10^6)$. Каждая из следующих n строк содержит по три целых числа: p_i, q_i и s_i $(1 \le p_i, q_i, s_i \le 1000)$.

Формат выходного файла

Первая строка выходного файла должна содержать одно вещественное число: w — общую стоимость подготовки студентов к операции. Вторая строка должна содержать одно вещественное число g — продолжительность групповой тренировки. Третья строка должна содержать n вещественных чисел — продолжительности личных тренировок студентов. При проверке вашего решения будет использоваться сравнение вещественных чисел с точностью 10^{-6}.

Пример

academy.in	academy.out
2 50 13 10 10 2 3 5 1	59.0 3.0 0.4 0.0

Задача B. Сумма делителей — 2

Имя входного файла:	`divsum2.in`
Имя выходного файла:	`divsum2.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Пусть x — натуральное число. Назовем y его *делителем*, если $1 \le y \le x$ и остаток от деления x на y равен нулю.

Задано число x. Найдите сумму его делителей, делящихся на каждое из простых чисел, на которое делится x.

Формат входного файла

Первая строка входного файла содержит целое число x ($1 \le x \le 10^{18}$). Все простые делители числа x не превосходят тысячу.

Формат выходного файла

В выходной файл выведите ответ на задачу.

Примеры

divsum2.in	divsum2.out
12	18
239	239

Задача C. Дельта, Каппа, Лямбда

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

Рассмотрим связный неориентированный граф G.

Степень вершины v, обозначаемая как $\deg v$ — это количество ребер, концом которых является эта вершина. Минимальная степень вершины в графе G обозначается как $\delta(G)$.

Вершинной связностью графа G, обозначаемой как $\kappa(G)$, называется минимальное число вершин, которые необходимо удалить из графа, чтобы он содержал как минимум две компоненты связности. Для полного графа K_n положим $\kappa(K_n) = n - 1$.

Реберной связностью графа G, обозначаемой как $\lambda(G)$, называется минимальное количество ребер, которые необходимо удалить из графа, чтобы он содержал как минимум две компоненты связности. Для графа с одной вершиной без ребер K_1 положим $\lambda(K_1) = 0$.

По заданным δ, κ и λ постройте граф, для которого $\delta(G) = \delta$, $\kappa(G) = \kappa$ и $\lambda(G) = \lambda$, либо определите, что такого графа не существует.

Формат входного файла

Входной файл содержит три целых числа: δ, κ и λ ($1 \le \delta, \kappa, \lambda \le 10$).

Формат выходного файла

Первая строка выходного файла должна содержать два целых числа: n и m — количество вершин и ребер в построенном графе, соответственно. Следующие m строк должны описывать ребра графа. Граф не должен содержать петель и кратных ребер. В графе должно быть не более 50 вершин.

Если искомого графа не существует, выведите на первой строке выходного файла два нуля.

Пример

dkl.in	dkl.out
2 2 2	3 3 1 2 2 3 1 3
1 1 2	0 0

Задача D. Купол

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

3141-й год. Люди уже давно колонизировали Луну. Однако иногда возникают небольшие трудности. Например, недавно на поверхность Луны упало два метеорита. В результате их падения образовалось два кратера, имеющих форму окружностей (поврехность Луны в этой задаче можно считать плоской). Центр первого кратера находится в точке (x_1, y_1), центр второго — в точке (x_2, y_2). Их радиусы равны r_1 и r_2 соответственно. При этом кратеры не имеют общих точек.

Кратеры оказались достаточно глубокими, поэтому было решено их неким образом закрыть. Для этого было решено возвести полусферический купол, на поверхности которого разместить солнечные батареи (нельзя же допустить, чтобы пропадало так много свободного места). Разумеется, чем больше радиус основания купола, тем больше требуется ресурсов и времени на его возведение. Поэтому требуется построить купол с минимальным радиусом основания.

Необходимо написать программу, которая по данным о расположении кратеров найдет минимальный радиус основания купола и положение центра купола.

Формат входного файла

Входной файл содержит шесть чисел: x_1, y_1, r_1 и x_2, y_2, r_2. Все числа во входном файле целые и не превосходят 10000 по абсолютному значению. Радиусы кратеров — положительные числа.

Формат выходного файла

В выходной файл выведите три числа: R, X, Y — соответственно минимальный радиус основания купола и координаты центра основания купола. Числа выводите с точностью не хуже 10^{-4}.

Примеры

dome.in	dome.out
0 0 1 2 0 1	2 1 0

Задача E. Произведение

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

Задано некоторое n-значное натуральное число x. Найдите произведение всех чисел, получающихся перестановкой цифр числа x. В выходной файл выведите остаток от деления этого произведения на число 366239.

Если одно и то же число получается с помощью нескольких перестановок, то его необходимо учитывать соответствующее число раз. Кроме этого, в получающихся с помощью перестановок числах могут быть ведущие нули.

Формат входного файла

Входной файл содержит целое число x ($1 \le x \le 10^9$). Оно задано в десятичной системе счисления и не содержит ведущих нулей.

Формат выходного файла

В выходной файл выведите остаток от деления искомого произведения на число 366239.

Примеры

product.in	product.out
12	252
11	121
10	10

Задача F. Голодный ферзь наносит ответный удар

Имя входного файла:	`queen2.in`
Имя выходного файла:	`queen2.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	64 мегабайта

Голодный шахматный ферзь стоит на поле $(0, 0)$ бесконечной шахматной доски. Также на доске расположено n пешек, пронумерованных от 1 до n, i-я пешка стоит на поле (x_i, y_i).

Ферзь планирует побить как можно больше пешек. При этом ферзь должен бить пешки по порядку: сначала первую, затем вторую, и т.д. Все ходы ферзя должны удовлетворять правилам шахмат — он должен ходить по горизонтали, вертикали или диагонали. Ферзь должен брать пешку каждым ходом. Не разрешается перепрыгивать через пешки. Взятая пешка снимается с доски. Пешки не перемещаются.

Выясните, какое максимальное количество пешек ферзь может побить.

Формат входного файла

Первая строк входного файла содержит n — количество пешек ($1 \le n \le 100\,000$). Следующие n строк содержат координаты пешек (координаты не превышают по модулю 10^9).

На поле $(0, 0)$ пешки нет, никакие две пешки не находятся на одном и том же поле.

Формат выходного файла

Выведите одно число — количество пешек, которые может побить ферзь.

Пример

queen2.in	queen2.out
4 0 2 1 1 1 -3 1 0	2

Задача G. Разделяемые разбиения

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

Рассмотрим прямоугольник, состоящий из $m \times n$ единичных квадратиков. Мы можем разбить его на две части, раскрасив некоторые его единичные квадратики в черный, а некоторые — в белый цвет, причем так, что и множество черных и множество белых квадратиков связно (множество называется связным если от любого квадратика этого множества можно дойти до любого другого, переходя с квадратика на соседний, имеющий с ним общее ребро).

Назовем такое разбиение *разделяемым*, если можно переместить белую и черную части таким образом, чтобы они находились сколь угодно далеко, непрерывно перемещая их без наложений. Например, разбиение на рисунке (а) является разделяемым, а разбиение на рисунке (б) — нет.

(а) Разделяемое разбиение. (б) Неразделяемое разбиение.

Выведите число разделяемых разбиений прямоугольника. Разбиения, которые различаются цветами частей, считаются одинаковыми. Обе части должны быть непусты.

Формат входного файла

Входной файл содержит числа m и n ($1 \le m, n \le 50$).

Формат выходного файла

Выведите одно число — количество разделяемых разбиений прямоугольника $m \times n$.

Пример

separable.in	separable.out
2 2	6
4 4	470

Задача H. Пастух

Имя входного файла:	`shepherd.in`
Имя выходного файла:	`shepherd.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

Антон — пастух. У него есть n овец, которые спокойно жуют травку на пастбище. А Антон сидит себе на холме, играет на дудочке, да наблюдает за своими овцами.

Но в соседнем лесу водятся злые волки! Овцы могут быть в опасности! Поэтому Антон очень внимательно следит, чтобы его овцы были в полной безопасности. Он хотел бы сидеть так, чтобы угол, под которым он видит овец, был как можно меньше. Разумеется, угол, под которым но видит на овец не должен превышать 180 градусов.

Но к сожалению, зрение у Антона довольно среднее, поэтому он плохо видит овцу, если она находится на расстоянии более D от него. Поскольку он должен хорошо видеть своих овец, никакая овца не должна быть дальше чем D от Антона.

Чтобы овцы не нервничали от присутствия человека, Антон не должен быть ближе d ни к какой из своих овец.

По информации о расположении овец, величинам D и d, определите, где следует находится Антону, чтобы угол, под которым он видел овец был как можно меньше (и не превышал 180 градусов).

Формат входного файла

Первая строка входного файла содержит три целых числа: n, d и D ($3 \le n \le 10$, $1 \le d \le D \le 10^3$).

Следующие n строк содержат по два целых числа — координаты овец. Координаты не превышают 10^3 по абсолютной величине. Никакие три овцы не находятся на одной прямой.

Формат выходного файла

Выведите два вещественных числа — координаты точки, где следует сидеть Антону, чтобы минимизировать угол, под которым он видит овец. Если решения не существует, выведите слово “`impossible`”.

Проверяющая программа будет использовать точность 10^{-5} для сравнения вещественных чисел. Выводите как можно больше знаков после десятичной точки.

Пример

shepherd.in	shepherd.out
4 1 5 0 0 1 0 0 1 1 1	4.9749371855331 0.5

Задача I. Разноцветная башня

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

Сережа играет с разноцветными кубиками — строит из них башню. Каждый кубик кроме самого большого он кладет на больший его. Самый большой кубик он ставит на пол в своей комнате.

Разумеется, при построении башни Сережа может использовать не все кубики. Однако он хочет, чтобы получившаяся башня была как можно более высокой. Для этого он хочет использовать при ее построении все кубики. К счастью, среди кубиков нет двух, имеющих одинаковый размер.

Кроме размера, каждый кубик характеризуется цветом — красным, синим или зеленым. Сережа хочет узнать, какая площадь поверхности построенной им башни будет окрашена в каждый из цветов.

Напишите программу, которая вычисляет ответ на сережин вопрос по описанию набора кубиков.

Формат входного файла

Первая строка входного файла содержит целое число n — количество кубиков ($1 \le n \le 50000$). Каждая из последующих n строк содержит описание одного кубика.

Описание кубика состоит из целого числа l ($1 \le l \le 10^7$) — длины его ребра и символа (`R`, `G`, `B`), обозначающего его цвет.

Длины ребер всех кубиков различны.

Формат выходного файла

В выходной файл выведите для каждого цвета площадь поверхности башни, имеющей такой цвет. Следуйте формату выходных данных, приведенному в примерах.

Примеры

tower.in	tower.out
3 1 R 2 G 3 B	R - 5 G - 19 B - 41