Задача A. Веревочки

Имя входного файла:	`ropes.in`
Имя выходного файла:	`ropes.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

С утра шел дождь, и ничего не предвещало беды. Но к обеду выглянуло солнце, и в лагерь заглянула СЭС. Пройдя по всем домикам и корпусам, СЭС вынесла следующий вердикт: бельевые веревки в жилых домиках не удовлетворяют нормам СЭС. Как выяснилось, в каждом домике должно быть ровно по одной бельевой веревке, и все веревки должны иметь одинаковую длину. В лагере имеется N бельевых веревок и K домиков. Чтобы лагерь не закрыли, требуется так нарезать данные веревки, чтобы среди получившихся веревочек было K одинаковой длины. Размер штрафа обратно пропорционален длине бельевых веревок, которые будут развешены в домиках. Поэтому начальство лагеря стремиться максимизировать длину этих веревочек.

Формат входных данных

В первой строке заданы два числа — N ($1 \leqslant N \leqslant 10001$) и K ($1 \leqslant K \leqslant 10001$). Далее в каждой из последующих N строк записано по одному числу — длине очередной бельевой веревки. Длина веревки задана в сантиметрах. Все длины лежат в интервале от 1 сантиметра до 100 километров включительно.

Формат выходных данных

В выходной файл следует вывести одно целое число — максимальную длину веревочек, удовлетворяющую условию, в сантиметрах. В случае, если лагерь закроют, выведите 0.

Примеры

ropes.in	ropes.out
4 11 802 743 457 539	200

Задача B. Корень кубического уравнения

Имя входного файла:	`cubroot.in`
Имя выходного файла:	`cubroot.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Дано кубическое уравнение $ax^3 + bx^2 + cx + d = 0$ ($a \neq 0$). Известно, что у этого уравнения есть ровно один корень. Требуется его найти.

Формат входных данных

Во входном файле через пробел записаны четыре целых числа: $-1000 \leqslant a, b, c, d \leqslant 1000$.

Формат выходных данных

Выведите единственный корень уравнения с точностью не менее 5 знаков после десятичной точки.

Примеры

cubroot.in	cubroot.out
1 -3 3 -1	1
-1 -6 -12 -7	-1.000000

Задача C. Дремучий лес

Имя входного файла:	`forest.in`
Имя выходного файла:	`forest.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Чтобы помешать появлению СЭС в лагере, администрация ЛКШ перекопала единственную дорогу, соединяющую "Берендеевы поляны" с Судиславлем, теперь проехать по ней невозможно. Однако, трудности не остановили инспекцию, хотя для СЭС остается только одна возможность — дойти до лагеря пешком. Как известно, Судиславль находится в поле, а "Берендеевы поляны" — в лесу.

- Судиславль находится в точке с координатами $(0, 1)$.
- "Берендеевы поляны" находятся в точке с координатами $(1, 0)$.
- Граница между лесом и полем — горизонтальная прямая $y = a$, где a — некоторое число $(0 \leqslant a \leqslant 1)$.
- Скорость передвижения СЭС по полю составляет V_p, скорость передвижения по лесу — V_f. Вдоль границы можно двигаться как по лесу, так и по полю.

Администрация ЛКШ хочет узнать, сколько времени у нее осталось для подготовки к визиту СЭС. Она попросила вас выяснить, в какой точке инспекция СЭС должна войти в лес, чтобы дойти до "Берендеевых полян" как можно быстрее.

Формат входных данных

В первой строке входного файла содержатся два положительных целых числа — V_p и V_f $(1 \leqslant V_p, V_f \leqslant 10^5)$. Во второй строке содержится единственное вещественное число — координата по оси Oy границы между лесом и полем a $(0 \leqslant a \leqslant 1)$

Формат выходных данных

В единственной строке выходного файла выведите вещественное число с точностью не менее 7 знаков после запятой — координата по оси Ox точки, в которой инспекция СЭС должна войти в лес.

Примеры

forest.in	forest.out
5 3 0.4	0.783310604
5 5 0.5	0.500000000

Задача D. Для любителей статистики

Имя входного файла:	`queries.in`
Имя выходного файла:	`queries.out`
Ограничение по времени:	2 секунды (4 секунды для Python 3)
Ограничение по памяти:	64 мегабайта

Вы никогда не задумывались над тем, сколько человек за год перевозят трамваи города с десятимиллионным населением, в котором каждый третий житель пользуется трамваем по два раза в день?

Предположим, что на планете Земля n городов, в которых есть трамваи. Любители статистики подсчитали для каждого из этих городов, сколько человек перевезено трамваями этого города за последний год. Из этих данных была составлена таблица, в которой города были отсортированы по алфавиту. Позже выяснилось, что для статистики названия городов несущественны, и тогда их просто заменили числами от 1 до n. Поисковая система, работающая с этими данными, должна уметь быстро отвечать на вопрос, есть ли среди городов с номерами от l до r такой, что за год трамваи этого города перевезли ровно x человек. Вам предстоит реализовать этот модуль системы.

Формат входных данных

В первой строке дано целое число n, $0 < n < 70\,000$. В следующей строке приведены статистические данные в виде списка целых чисел через пробел, i-е число в этом списке — количество человек, перевезенных за год трамваями i-го города. Все числа в списке положительны и не превосходят $10^9 - 1$. В третьей строке дано количество запросов q, $0 < q < 70\,000$. В следующих q строках перечислены запросы. Каждый запрос — это тройка целых чисел l, r и x, записанных через пробел ($1 \leqslant l \leqslant r \leqslant n$, $0 < x < 10^9$).

Формат выходных данных

Выведите строку длины q, в которой i-й символ равен `1`, если ответ на i-й запрос утвердителен, и `0` в противном случае.

Примеры

queries.in	queries.out
5 123 666 314 666 434 5 1 5 314 1 5 578 2 4 666 4 4 713 1 1 123	10101

Задача E. Поезда

Имя входного файла:	`trains.in`
Имя выходного файла:	`trains.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

В связи с участившимся числом аварий на железнодорожной ветке Кострома–Судиславль, руководство железной дороги решило изменить график движения поездов. Тщательный анализ состояния железнодорожного полотна показал, что оптимальным является следующий график движения поездов с учетом остановок на станциях: сначала поезд идет на протяжении T_1 минут со скоростью V_1 метров в минуту, затем T_2 минут со скоростью V_2 метров в минуту, ..., наконец, T_N минут со скоростью V_N метров в минуту. В течение некоторых интервалов поезд может стоять (скорость равна 0).

По действующей инструкции обеспечения безопасности движения поездов расстояние между локомотивами двух следующих друг за другом поездов должно быть не менее L метров. Определите минимально допустимый интервал в минутах между отправлениями поездов, позволяющий им двигаться по этому графику без опасного сближения.

Формат входных данных

В первых двух строках входного файла содержится два натуральных числа, задающие минимально допустимое расстояние L и количество участков пути N ($100 \leqslant L \leqslant 10\,000$, $1 \leqslant N \leqslant 1000$). Далее следует N пар целых чисел T_i и V_i, задающих график движения поездов ($1 \leqslant T_i \leqslant 1000$, $0 \leqslant V_i \leqslant 1000$).

Формат выходных данных

В выходной файл необходимо вывести искомый интервал между отправлениями поездов в минутах, не менее чем с тремя верными знаками после десятичной точки.

Примеры

trains.in	trains.out
1000 4 10 0 30 80 15 0 20 100	27.500