Задача A. Проблема сапожника

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

В некоей воинской части есть сапожник. Рабочий день сапожника длится K минут. Заведующий складом оценивает работу сапожника по количеству починенной обуви, независимо от того, насколько сложный ремонт требовался в каждом случае. Дано n сапог, нуждающихся в починке. Определите, какое максимальное количество из них сапожник сможет починить за один рабочий день.

Формат входных данных

В первой строке вводятся числа K (натуральное, не превышает 1000) и n (натуральное, не превышает 500). Затем идет n чисел — количество минут, которые требуются, чтобы починить i-й сапог (времена — натуральные числа, не превосходят 100).

Формат выходных данных

Выведите одно число — максимальное количество сапог, которые можно починить за один рабочий день.

Примеры

cobbler.in	cobbler.out
10 3 6 2 8	2
3 2 10 20	0

Задача B. Мороженое

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	512 мегабайт

Недавно Макару подарили n стаканчиков мороженого, i-й из которых содержит a_i грамм лакомства. Сегодня он решил все их съесть.

Когда Макар начнет есть, он достанет мороженое из морозилки, поэтому в тот же момент мороженое в каждом стаканчике начнет таять. Каждое мороженое непрерывно тает со скоростью v грамм в секунду. Растаявшее мороженое Макара не интересует, но при этом никак ему и не мешает: можно считать, что мороженое тает снизу, а Макар ест его сверху. Сам Макар ест мороженое непрерывно со скоростью u грамм в секунду. В каждый момент времени Макар может есть только одно мороженое. При этом он может начать с любого стаканчика и в любой момент времени может переключаться с одного стаканчика на другой (это действие происходит мгновенно).

Однажды начав, Макар уже не сможет остановиться и будет есть до тех пор, пока еще остается нерастаявшее мороженое. Но Макар знает, что есть много сладкого — вредно! Поэтому он хочет есть мороженое таким образом, чтобы минимизировать суммарную массу съеденного мороженого к моменту, когда его больше не останется.

Помогите Макару! Выведите минимальную суммарную массу мороженого, которую он может съесть при заданных условиях.

Формат входных данных

В первой строке задано 3 целых числа: n, v и u ($1 \leqslant n \leqslant 3 \cdot 10^5$, $1 \leqslant v, u \leqslant 10^9$) — количество стакничков мороженого, скорость таяния мороженого и скорость поедания Макаром мороженого, соответственно.

Во второй строке n целых чисел a_i ($1 \leqslant a_i \leqslant 10^9$) — масса мороженого в граммах в каждом из стаканчиков.

Формат выходных данных

Выведите одно число — минимальное количество мороженого, которое может съесть Макар, с относительной или абсолютной погрешностью не более 10^{-6}.

Примеры

стандартный ввод	стандартный вывод
1 1 2 90	60.0000000000
2 1 1 30 20	16.6666666667

Задача C. Протеиновый батончик

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

Качок Виталий, возвращаясь с очередной тренировки, по привычке заскочил в магазин «Жму 120». Сегодня он захотел попробовать что-нибудь новое, например, протеиновый батончик. Подойдя к прилавку, он взял один и заметил, что сзади написан состав.

Виталий знает, что ингредиенты любого продукта указаны в порядке невозрастания массовой доли. Он также знает, что батончик является полезным, если i-го ингредиента в нём не менее l_i грамм и не более r_i грамм. При этом масса самого батончика (то есть суммарная масса всех ингредиентов) равна в точности x.

Зная массу батончика x и границы l_i и r_i для каждого из ингредиентов, проверьте, может ли данный батончик быть полезным, или Виталию точно не следует его брать.

Формат входных данных

В первой строке входных данных даны два целых числа n и x ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant x \leqslant 10^9$) — количество ингредиентов и масса батончика соответственно. В следующих n строках вводятся пары l_i, r_i ($1 \leqslant l_i \leqslant r_i \leqslant 10^9$) — границы массы i-го ингредиента, при которых батончик является полезным.

Формат выходных данных

В единственной строке входных данных выведите «YES» (без кавычек), если такой батончик существует и «NO» (без кавычек) в противном случае.

Примеры

protein.in	protein.out
2 5 2 5 1 4	YES
2 10 2 5 1 4	NO

Замечание

В первом тесте существует полезный батончик, например, если первого ингредиента 3 грамма, а второго 2. Этот батончик корректный, так как, во-первых, суммарный вес равен $3 + 2 = 5$, во-вторых, масса первого ингредиента не меньше, чем масса второго ($3 \geqslant 2$), а в-третьих, 3 и 2 лежат в нужных массовых границах ($2 \leqslant 3 \leqslant 5$, $1 \leqslant 2 \leqslant 4$).

Задача D. Подготовка олимпиады

Имя входного файла:	`prepare-contest.in`
Имя выходного файла:	`prepare-contest.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

До олимпиады младших параллелей осталось всего пара часов, а члены жюри ещё только проснулись! Теперь им нужно срочно составить контест. В распоряжении у них есть n задач различной сложности, из которых необходимо выбрать несколько задач для олимпиады.

Про некоторые из имеющихся n задач жюри знает их сложность — для каждой из них это целое число от 1 до m ($m \leqslant n$). Остальные задачи *универсальны*, то есть их сложность можно произвольно варьировать от 1 до m (например, меняя ограничения на размер входных данных).

Жюри хочет составить контест следующим образом: всего в нём должно быть m задач, причём первая задача должна иметь сложность 1, вторая задача — сложность 2, ..., m-я задача — сложность m. В качестве задачи сложности k можно использовать любую из *универсальных* задач, каждую не более одного раза.

Без вашей помощи олимпиада не будет готова в срок! Напишите программу, определяющую, возможно ли составить контест согласно описанным условиям, и если возможно, выводит упорядоченный по сложности список задач, из которых должен быть составлен контест. Если таких способов существует несколько, необходимо вывести лексикографически минимальный из них.

Лексикографическое сравнение двух различных наборов из m задач производится следующим образом. Выбирается минимальная сложность k такая, что в двух наборах даны разные задачи сложности k — пусть это задачи с названиями S и T из первого и второго набора соответственно. Затем лексикографически сравниваются названия этих двух задач. Если S оказывается лексикографически меньше, чем T, то лексикографически меньшим считается первый набор задач; иначе — второй.

Напомним, как лексикографически сравниваются различные строки S и T. Если одна из них является префиксом другой, то более короткая считается лексикографически меньшей. Иначе находится первая позиция, в которой они различаются, и меньшей считается та строка, в которой на этой позиции стоит буква, идущая в алфавитном порядке раньше, чем в другой строке.

Формат входных данных

В первой строке входных данных заданы целые числа n и m ($1 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant n$).

В следующих n строках заданы задачи, имеющиеся в распоряжении жюри. В $(i+1)$-й строке входных данных записана информация об i-й задаче: название задачи, затем, через пробел, сложность задачи. Название задачи — это строка длины не более 10, состоящая только из строчных латинских букв. Сложность задачи — целое число от 1 до m для неуниверсальных задач; если же задача *универсальная*, на месте сложности записан 0.

Гарантируется, что все названия задач различны.

Формат выходных данных

Если составить контест, выполнив необходимые требования, невозможно, выведите «`-1`». Иначе выведите m строк: на i-й строке название i-й по сложности задачи в контесте.

Примеры

prepare-contest.in	prepare-contest.out
5 3 aplusb 2 binary 1 count 0 derive 3 lower 0	binary aplusb count
5 3 aplusb 2 binary 1 count 1 derive 1 lower 1	-1

Задача E. Код Хаффмана

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

Заданы числа $p_1, p_2, \ldots, p_n$.

Предположив, что имеется текст, содержащий p_1 символов c_1, p_2 символов c_2, и т. д., постройте код Хаффмана и найдите суммарное число битов, необходимое для кодирования такого текста.

Формат входных данных

Первая строка входного файла содержит число n ($2 \leqslant n \leqslant 1000$). Вторая строка содержит n целых чисел $p_1, p_2, \ldots, p_n$ ($1 \leqslant p_i \leqslant 10^9$).

Формат выходных данных

Выведите одно число — число битов, необходимое для кодирования текста с заданным во входном файле количеством вхождений каждого символа.

Пример

huffman.in	huffman.out
10 1 2 3 4 5 6 7 8 9 10	173

Задача F. Rejecter-9000

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Как вам всем известно, преподавателям ЛКШ лень проверять посылки своих учеников. Поэтому они изобрели специальный скрипт Rejecter-9000, который позволяет им ускорить этот процесс.

Скрипт для каждого школьника рассматривает все его посылки, ожидающие проверки (Pending Review). Пока у школьника есть хотя бы одна непроверенная посылка, и количество таких посылок чётно, скрипт отклоняет с Reject-ом половину из них. Например, если у школьника было 6 непроверенных посылок, то после запуска скрипта останется только 3, а если их было 8, то только одна.

В параллели Y учится n человек, а сегодняшний контест преподаватели перегробили, и поэтому в нем очень много задач. Вернувшись с обеда, Афанасий обнаружил, что i-й ученик сдал a_i задач, которые ожидают проверки. Остальные задачи у учеников совсем не идут, и никакой ученик не может решить никакую новую задачу без помощи Афанасия. В целом, у Афанасия есть три опции:

- Он может помочь школьнику отдебажить его код по нерешённой задаче. После этого школьник сдает задачу, и количество его непроверенных посылок увеличивается на единицу. После обеда дети усердно ботают, поэтому в любой момент времени у любого школьника есть задача, которую можно отдебажить.
- Если у ученика есть хотя бы одна непроверенная посылка, то Афанасий может проверить одну из них и поставить ей нужный вердикт (OK или Rejected). Количество непроверенных посылок тем самым уменьшается на один.
- Он может запустить скрипт Rejecter-9000. После его выполнения у каждого ученика количество непроверенных посылок, если оно ненулевое, сократится в 2^k раз, где k — максимальная степень двойки, на которую делится это число.

Rejecter-9000 написан на медленном, но стабильном языке Bidon, и работает всегда одну минуту. Дебаг и проверка решений школьников у Афанасия занимают также по одной минуте его ценного времени. Афанасий очень беспокоится о своём свободном времени, потому что каждая лишняя минута — это минута сна. Помогите ему и скажите, за какое минимальное время Афанасий может уменьшить суммарное количество непроверенных посылок до нуля.

Формат входных данных

В первой строке дано число n ($1 \leqslant n \leqslant 100\,000$) — количество учеников в параллели Y. Во второй строке записано n чисел — количество непроверенных задач для каждого ученика ($0 \leqslant a_i \leqslant 10^9$).

Формат выходных данных

Выведите минимальное количество минут, за которое Афанасий может добиться отсутствия непроверенных посылок.

Примеры

стандартный ввод	стандартный вывод
3 1 5 7	6

Замечание

Вот одна из возможных последовательностей действий Афанасия в примере: проверка задач первого и второго школьника, помощь третьему школьнику, запуск скрипта, проверка задач второго и третьего школьника.

Задача G. Золотой песок

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

Во время ограбления магазина вор обнаружил N ящичков с золотым песком. В ящичке под номером i песок имеет общую стоимость v_i и вес w_i. Чтобы унести награбленное, вор использует рюкзак. Требуется определить наибольшую суммарную стоимость песка, который может унести грабитель, если грузоподъёмность рюкзака ограничена величиной W.

Из ящичков можно пересыпать любое количество песка — тогда отношение стоимости отсыпанного песка к стоимости всего ящичка будет равно отношению веса пересыпанного песка к весу всего ящичка.

Формат входных данных

В первой строке входного файла записано два числа — N и W ($1 \leqslant N \leqslant 10\,000$, $0 \leqslant W \leqslant 10^6$). Далее следуют N строк, в каждой из которых записано по два целых числа: в i-й строке записана стоимость v_i и вес w_i песка в i-м ящичке. Все числа неотрицательны и не превосходят 10^6.

Формат выходных данных

Выведите искомую максимальную стоимость.

Примеры

dust.in	dust.out
3 50 60 20 100 50 120 30	180.0

Задача H. Интернет

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

В этом году в ЛКШ появился общедоступный быстрый интернет. Теперь преподавателям приходится делиться драгоценными мегобитами со школьниками. Тем не менее, школьников всегда можно отключить от сети всего в один клик. Однако школьники быстро просекли эту тему и написали письмо в «МДДШИ» (Министерство по делам о доступе школьников в интернет)

Преподаватели смогли убедить министерство, что они не могут выделить школьникам больше, чем x Мб/с из-за необходимости поддержания учебного процесса. Теперь перед преподавателями стоит задача по одному отключить школьников от интернета. А так как они очень злы на них из-за внезапной проверки, они это будут делать с особым наслаждением. В итоге преподаватели хотят отключить максимальное количество школьников так, чтобы в момент удаления любого из них суммарное потребление интернета было больше, чем x, чтобы не получить штраф из МДДШИ. На данный момент в сети n школьников и каждый из них жрёт x условных единиц интернета.

Формат входных данных

В первой строке находится два числа. $1 \leqslant n \leqslant 10^5$ - количество школьников в сети и $1 \leqslant x \leqslant 10^{18}$ - установленная планка потребления Во второй строке находится n чисел. $1 \leqslant a_i \leqslant 10^9$ - количество трафика занимаемого i-тым школьником.

Формат выходных данных

В единственной строке выведите одно число - максимальное количество школьников, которых можно отключить таким методом.

Примеры

disconnect.in	disconnect.out
4 5 1 2 3 4	3
5 20 3 5 22 6 13	5
4 8 1 2 3 4	2

Задача I. Максимальная сумма «И»

Имя входного файла:	`andsum.in`
Имя выходного файла:	`andsum.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	256 мегабайт

Омар Вольдемар обожает шестнадцатеричные числа, а также очень любит операцию побитового «И», поэтому он наверняка обрадовался бы этой задаче.

Даны n целых неотрицательных чисел, заданных четырьмя цифрами в шестнадцатеричной системе счисления. Требуется разбить эти числа на два **непустых** множества так, чтобы сумма результата операции побитового «И» всех чисел первого множества и результата операции побитового «И» всех чисел второго множества была максимальна.

Формат входных данных

В первой строке входных данных записано число n ($2 \leqslant n \leqslant 1000$) — количество чисел, которые требуется разбить на два непустых множества. В последующих n строках представлены числа a_i в виде четырёх цифр в шестнадцатеричной системе счисления (возможно, с ведущими нулями).

Формат выходных данных

Выведите максимальную сумму результата операции побитового «И» для всех чисел первого множества и результата операции побитового «И» для всех чисел второго множества, которую можно получить при разбиении исходного множества на два. Ответ требуется вывести в шестнадцатеричной системе счисления без ведущих нулей.

Примеры

andsum.in	andsum.out
3 0023 031A 0121	33B
2 8000 8000	10000
3 0000 0000 0000	0

Замечание

В первом примере даны числа, которые в десятичной системе счисления записываются как 35, 794 и 289. Если отнести 794 к одной группе, а 35 и 289 к другой группе, то получатся числа 794 и 33, сумма которых 827, а если перевести в шестнадцатеричную систему — $33B$.

Задача J. Мороженое

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

Вдоль моря узкой полоской тянется пляж. В некоторых точках пляжа расположены ларьки с мороженым. В один прекрасный день не все мороженщики вышли на работу. Распределите мороженщиков по ларькам так, чтобы минимальное расстояние между мороженщиками было как можно больше. Так они будут меньше мешать друг другу.

Формат входных данных

В первой строке вводятся числа N ($2 < N < 10\,001$) — количество ларьков и K ($1 < K < N$) — количество мороженщиков, вышедших на работу. Во второй строке задаются N натуральных чисел в порядке возрастания — координаты ларьков (координаты не превосходят 10^9).

Формат выходных данных

Выведите одно число — минимальное расстояние между соседними ларьками в оптимальной расстановке.

Примеры

ice-cream.in	ice-cream.out
5 3 1 2 3 100 1000	99