Задача A. Закон Амдала

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

Параллельное программирование изучает методы построения программ, которые будут выполняться на нескольких процессорах. В результате решения одной из первых задач этого раздела информатики появился *закон Амдала*.

Задача Амдала формулировалась так. Имеется n процессоров и p процентов вычислений не могут выполняться параллельно. Во сколько раз быстрее можно выполнить вычисления по сравнению с одним процессором?

Например, если $n = 10, p = 50$, а на одном процессоре все вычисления выполняются за время t. Тогда первая половина вычислений(50%) будет выполнена за время $\frac{t}{2 \cdot 10}$, а вторая — за время $\frac{t}{2}$. Общее время вычислений в этом случае составит $\frac{t}{2} + \frac{t}{20} = \frac{11 \cdot t}{20}$, а ускорение по сравнению с одним процессором составит $\frac{20}{11}$ раза.

Если же $n = 10, p = 25$, и на одном процессоре все вычисления выполняются за время t. Тогда 75% вычислений будут выполнены за время $\frac{3 \cdot t}{4 \cdot 10}$, а оставшиеся 25% — за время $\frac{t}{4}$. Общее время вычислений в этом случае составит $\frac{t}{4} + \frac{3 \cdot t}{40} = \frac{13 \cdot t}{40}$, а ускорение по сравнению с одним процессором составит $\frac{40}{13}$ раза.

Даны числа n и p. Напишите программу, решающую задачу Амдала.

Формат входного файла

Входной файл содержит два целых числа n $(1 \le n \le 1000)$ и p $(0 \le p \le 100)$.

Формат выходного файла

В выходной файл выведите ответ на задачу с точностью не хуже 10^{-6}.

Примеры

amhdal.in	amhdal.out
10 50	1.818181818
10 25	3.076923077
1000 100	1.00000000000
1000 0	1000.000000000
239 30	3.301104972
777 55	1.816269285

Задача B. Биатлон

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

На Зимних Олимпийских Играх традиционно проводятся соревнования по биатлону. Как известно, этот вид спорта содержит лыжные гонки и стрельбу по мишеням из винтовки. На каждом огневом рубеже расположены 5 мишеней. Каждая из них имеет форму круга радиусом 10 см, а расстояния между центрами соседних мишеней одинаковы и равны 25 см. Центры мишеней при этом расположены на одной горизонтали.

Введем прямоугольную систему координат так, что начало координат расположено в центре самой левой мишени, ось Ox направлена вправо, а ось Oy — вверх. Таким образом, центры мишеней имеют координаты $(0, 0)$, $(25, 0)$, $(50, 0)$, $(75, 0)$ и $(100, 0)$.

Для информационного обеспечения проведения соревнований было решено разработать систему подсчета количества пораженных мишеней. Эта система по точкам, в которые попали пули после выстрелов спортсмена, должна определять количество пораженных мишеней. Мишень считается пораженной, если в нее попала хотя бы одна пуля (при этом, разумеется, если в мишень попали две или больше пуль, то попадание считается только один раз).

На спринтерской гонке на каждом огневом рубеже у спортсмена есть 5 пуль. Вам даны координаты точек, в которые попали пули после выстрелов спортсмена. Определите количество пораженных мишеней.

Формат входного файла

Входной файл содержит ровно пять строк: i-ая из них содержит два целых числа x_i и y_i — координаты точки, в которую попала пуля после i-ого выстрела спортсмена. Все числа во входном файле не превосходят 1000 по модулю.

Формат выходного файла

В выходной файл выведите единственное число — количество пораженных мишеней.

Примеры

biathlon.in	biathlon.out
0 0 25 0 50 0 75 0 100 0	5
0 0 0 0 0 0 75 0 100 0	3

Задача C. Карточки

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

Вася и Петя играют в следующую игру. Вася кладет на стол два ряда карточек. Первый ряд состоит из n карточек, на каждой из которых написано некоторое число a_i. Второй ряд состоит из n карточек, на каждой из которых написано некоторое число b_i.

Пете требуется переупорядочить карточки первого ряда так, чтобы на столе получилось два одинаковых ряда карточек. За одну секунду Петя может поменять местами i-ую и $i+1$-ую $(1 \le i \le n-1)$ карточки первого ряда.

Помогите Пете переупорядочить карточки затратив на это минимальное время.

Формат входного файла

Первая строка входного файла содержит целое число n $(1 \le n \le 100\,000)$. Вторая строка содержит n целых чисел a_i. Третья строка содержит n целых чисел b_i. Все числа по абсолютной величине не превосходят 10^6.

Формат выходного файла

В первой строке выходного файла выведите количество секунд, за которые Петя сможет переупорядочить карточки требуемым образом. Если переупорядочить карточки требуемым образом невозможно, выведите единственное число -1.

Примеры

cards.in	**cards.out**

`5`
`3 2 3 4 5`
`5 4 3 2 3`
	`7`

`8`
`1 2 1 2 1 2 1 2`
`2 1 2 1 2 1 2 2`
	`-1`

Задача D. Шрифты

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

Третий за неделю чек по заправке картриджа для принтера в бухгалтерии приняли без особого удовольствия. Судя по всему, надо серьезно подойти к вопросу экономного использования тонера — на учет должна быть поставлена каждая буква. Вам поручена реализация утилиты, которая будет обрабатывать HTML-документы старых версий и подсчитывать в них количество символов каждого размера. HTML-документ состоит из обычного текста и тегов - управляющих элементов, заключенных в угловые скобки. Для управления размером шрифта используется тег *font* с атрибутом *size*: . Если x — натуральное число, то размер шрифта устанавливается равным x. Кроме того, x может иметь вид $+y$ или $-y$, где y — натуральное число. В этом случае размер шрифта соответственно увеличивается или уменьшается на y. Действие тега заканчивается с появлением соответствующего закрывающего тега . Все остальные теги вы не должны обрабатывать. Можете считать, что теги не содержат лишних пробельных символов. По умолчанию размер шрифта равен 10, и изменения не будут выводить его из интервала от 1 до 50.

Формат входного файла

Входной файл содержит описание HTML-документа, по длине не превосходящее 5000 символов.

Формат выходного файла

Для каждого встречающегося в документе размера шрифта выведите на отдельной строке через пробел его размер и количество соответствующих ему непробельных (с ASCII кодами не равными 9, 10, 13 и 32) символов в порядке возрастания размера.

Примеры

font.in	**font.out**
<HTML> <BODY> Fonts: normal tiny bigger <p> as before very big smaller again ... </BODY> </HTML>	2 4 7 12 10 20 11 6 15 10

Задача E. Хеш-функция

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

В задачах поиска часто используются так называемые *хеш-функции*. Одним из важнейших классов хеш-функций являются так называемые *полиномиальные хеш-функции*.

Пусть дана строка $S = s_1 s_2 \ldots s_l$, состоящая из цифр от 0 до 9. Тогда значение *полиномиальной хеш-функции* $p(S, x, m)$ вычисляется следующим образом

$$p(S, x, m) = (\sum_{i=1}^{l} s_i x^{i-1}) \bmod m$$

($a \bmod b$ обозначает остаток от деления числа a на число b). Например, пусть $S = 0123$, тогда $p(S, 2, 5) = (0 \cdot 1 + 1 \cdot 2 + 2 \cdot 4 + 3 \cdot 8)\ mod\ 5 = 4$.

Вам даны множество из n строк $(S^{(1)}, S^{(2)}, \ldots, S^{(n)})$, каждая из которых состоит только из цифр от 0 до 9, и числа m и x. Необходимо найти количество таких пар (i, j), где $1 \le i, j \le n, i < j$, что $p(S^{(i)}, x, m) = p(S^{(j)}, x, m)$.

Формат входного файла

Первая строка входного файла содержит три целых числа: n ($1 \le n \le 30000$), m ($1 \le m \le 2000$), x ($1 \le x \le 100$). Далее идут n строк, каждая из которых содержит по одной строке из данного множества: 2-ая строка входного файла содержит $S^{(1)}$, 3-я — $S^{(2)}$, …, $(i+1)$-ая — $S^{(i)}$, …, $(n+1)$-ая — $S^{(n)}$. Длины $S^{(i)}$ не превосходят 100, $S^{(i)}$ непусты и состоят только из цифр от 0 до 9.

Формат выходного файла

Выведите в выходной файл одно число — ответ на задачу.

Примеры

hash.in	hash.out
8 3 8 1234 239 366 261 32890 43823490 382390 3043840	11
5 10 100 1 2 3 4 5	0

Задача F. Список

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

В наше время создатели офисных приложений стараются сделать все для удобства пользователя. Поэтому даже такая мелочь, как представление на экране списков чисел — например, для вывода номеров страниц, — должна быть тщательно проработана.

Вы должны реализовать функцию, которая по заданному набору целых чисел будет формировать строку, являющуюся его самым коротким текстовым представлением. Текстовое представление — строка, состоящая из разделенных запятыми чисел и диапазонов чисел вида «a`, ..., `b», которые используются для записи набора всех чисел от a до b. При этом все числа, входящие в строку должны быть упорядочены по возрастанию в том порядке, в котором они встречаются в строке.

Формат входного файла

В первой строке входного файла содержится целое число n ($1 \le n \le 1000$) — размер набора. Вторая строка содержит n задающих набор целых чисел, по абсолютной величине не превосходящих 10000, разделенные пробелами. Одно число может встречаться в этом описании несколько раз.

Формат выходного файла

В первой строке выходного файла запишите одно из кратчайших текстовых представлений заданного набора чисел. Следите за правильной расстановкой пробелов. Выходной файл в примере содержит ровно три пробела.

Примеры

list.in	**list.out**
`7` `1 3 5 -1 3 4 6`	`-1, 1, 3, ..., 6`

Задача G. Покупки

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

Во многих фирмах, занимающихся торговлей, существует должность менеджера по закупкам. Как известно, они занимаются тем, что по торговому плану, представляющему собой список наименований товаров, для каждого из которых указано необходимое количество, закупает указанные в нем товары на оптовых базах. Торговый план при этом составляется руководством компании. Иногда у менеджеров по закупкам возникает желание принести выгоду не только своей фирме, но и себе.

Только что, как раз после подписания очередного торгового плана на заказ n наименований товаров, открылась новая оптовая база. Как это часто бывает сразу после открытия, ее цены на многие товары ниже заложенных в план. Наверное, этим можно воспользоваться.

На закупку товаров были выделены деньги из расчета того, что все товары будут закупаться на старой оптовой базе. Менеджер хочет, воспользовавшись возможностью покупать товары на новой базе, потратить как можно меньше денег на закупку требуемого количества товаров (непотраченные деньги он, конечно, сможет забрать себе).

Чтобы не вызывать сильных подозрений, производить на новой базе закупки, суммарная запланированная стоимость которых была больше, чем d денежных единиц, не следует. Осталось только рассчитать, какие товары и в каком количестве следует закупать на новой базе, чтобы осталось как можно больше непотраченных денег.

Формат входного файла

Первая строка содержит четыре числа: n $(1 \le n \le 1000)$, d, а так же k_1 и k_2 $(1 \le k_1, k_2 \le 1000)$ — количества наименований товаров, имеющихся на открытых ранее и новой базе соответственно. После этого идут n строк, каждая из которых содержит название товара в плане и его количество (положительное вещественное число). За ними следуют два блока из k_1 и k_2 строк соответственно, отделенные от предыдущего и разделенные между собой переводом строки — наименования товаров на базах и цены за единицу товара соответственно. Все цены являются положительными числами, даже на новой базе.

Названия товаров состоят из не более, чем 100 латинских букв и символов подчеркивания, при этом регистр букв не учитывается. Вещественные числа заданы не более чем с двумя знаками после десятичной точки и по величине не превосходят 10^6. Гарантируется, что все товары из плана можно купить на старой базе.

Формат выходного файла

В выходной файл выведите n вещественных чисел, по одному на строке, задающих количество соответствующего товара, закупаемого на новой базе. На i-ой строке выведите количество товара, идущего i-ым в плане. Ошибки менее 0.01 будут игнорироваться.

Примеры

shopping.in	shopping.out
4 11.0 5 4 sugar 0.5 powdered_milk 7 flour 8 salt 3 salt 0.5 flour 1 sugar 20 powdered_milk 3 cinnamon 8 sugar 10 flour 2 salt 0.4 liquid_hydrogen 10000	0.5000 0.0000 0.0000 2.0000

Задача H. Табло

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

На хоккейном стадионе в одном большом городе расположено большое прямоугольное табло. Оно имеет n строк и m столбцов (то есть состоит из $n \times m$ ячеек). Во время хоккейного матча это табло служит для отображения счета и времени, прошедшего с начала тайма, а в перерывах на нем показывают различную рекламу.

В связи с этим возникла задача проверки возможности показа на этом табло определенной рекламной заставки. Заставка также, как и табло, имеет размер n строк на m столбцов. Каждая из ячеек заставки окрашена в один из четырех цветов — трех основных: красный — `R`, зеленый — `G`, синий — `B` и черный — `.`(точка).

Каждая из ячеек табло характеризуется своими цветопередаточными возможностями. Любая из ячеек табло может отображать черный цвет — это соответствует тому, что на нее вообще не подается напряжение. Также каждая из ячеек может отображать некоторое подмножество множества основных цветов. В этой задаче эти подмножества будут кодироваться следующим образом:

- `0` — ячейка может отображать только черный цвет;
- `1` — ячейка может отображать только черный и синий цвета;
- `2` — ячейка может отображать только черный и зеленый цвета;
- `3` — ячейка может отображать только черный, зеленый и синий цвета;
- `4` — ячейка может отображать только черный и красный цвета;
- `5` — ячейка может отображать только черный, красный и синий цвета;
- `6` — ячейка может отображать только черный, красный и зеленый цвета;
- `7` — ячейка может отображать только черный, красный, зеленый и синий цвета.

Напишите программу, которая по описанию табло и заставки определяет, возможно ли на табло отобразить эту заставку.

Формат входного файла

Первая строка входного файла содержит целые числа n и m ($1 \le n, m \le 100$). Далее идут n строк по m символов каждая — описание заставки. Каждый из символов описания заставки принадлежит множеству `{R, G, B, .}` . Их значения описаны выше.

После этого идет описание табло. Оно содержит n строк по m чисел, разделенных пробелами. Значения чисел описаны выше.

Формат выходного файла

В выходной файл выведите `YES`, если на табло возможно отобразить заставку и `NO` — в противном случае.

Примеры

tableau.in	tableau.out
3 3 .GB R.B RG. 0 1 2 3 4 5 6 7 0	NO
2 3 RGB .G. 7 7 7 7 7 7	YES