# Задача A. Большой линейный коллайдер

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

Группа ученых работает в международной научной лаборатории, которая занимается исследованиями поведения элементарных частиц в установке для экспериментов «Большой линейный коллайдер» (БЛК). Установка БЛК представляет собой прямую, в некоторых точках которой размещаются частицы, которые могут перемещаться вдоль прямой.

В очередном эксперименте в БЛК размещаются n частиц, каждая из которых представляет собой либо отрицательно заряженную частицу — электрон $e^-$, либо положительно заряженную частицу — позитрон $e^+$. В эксперименте $i$-я частица исходно размещается в точке с координатой $x_i$. После начала эксперимента в результате работы БЛК частицы начнут перемещаться в разные стороны вдоль прямой: $e^-$ частицы перемещаются по направлению уменьшения координаты, а $e^+$ частицы — по направлению увеличения координаты. Абсолютные величины скоростей всех частиц одинаковы и равны 1. Если в процессе перемещения частицы $e^-$ и $e^+$ оказываются в одной точке, то они взаимодействуют и обе исчезают, при этом они не влияют на дальнейшее поведение остальных частиц.

Ученые выбрали $m$ различных моментов времени $t_1, t_2, \ldots, t_m$, для каждого из которых их интересует, какое количество частиц находится в БЛК непосредственно после каждого из этих моментов времени. Отсчет времени начинается с момента 0, когда частицы приходят в движение. Частицы, исчезнувшие в результате взаимодействия в момент времени $t_j$, не должны учитываться при подсчете количества частиц для этого момента времени. Требуется написать программу, которая по описанию исходного расположения и типов частиц, а также заданным моментам времени, определяет для каждого из моментов количество частиц, которое будет находиться в БЛК непосредственно после этого момента.

## Формат входных данных

Первая строка входного файла содержит число $n$ — количество частиц ($1 \leqslant n \leqslant 200\,000$). Последующие $n$ строк описывают частицы следующим образом: каждая строка содержит по два целых числа $x_i$ и $v_i$ — координату $i$-й частицы и ее тип соответственно ($-10^9 \leqslant x_1 < x_2 < \ldots < x_n \leqslant 10^9$, $v_i$ равно $-1$ или 1). Частица $e^-$ описывается значением $v_i = -1$, а частица $e^+$ описывается значением $v_i = 1$.

Следующая строка содержит целое число $m$ — количество моментов времени, которые выбрали ученые ($1 \leqslant m \leqslant 200\,000$). Последняя строка содержит $m$ целых чисел: $t_1, t_2, \ldots, t_m$ ($0 \leqslant t_1 < t_2 < \ldots < t_m \leqslant 10^9$).

## Формат выходных данных

Для каждого момента времени во входном файле требуется вывести одно число: количество частиц в БЛК непосредственно после этого момента.

## Примеры

| linear.in | linear.out |
|---|---|
| 4<br>-1 1<br>0 -1<br>1 1<br>5 -1<br>4<br>0 1 2 3 | 4<br>2<br>0<br>0 |

## Замечания

В приведенном примере в начальный момент в БЛК находятся 4 частицы: частица $e^+$ в точке –1, частица $e^-$ в точке 0, частица $e^+$ в точке 1 и частица $e^-$ в точке 5.

В момент времени 0.5 первая частица $e^+$ и первая частица $e^-$ сталкиваются в точке с координатой –0.5 и исчезают. В момент времени 1 оставшиеся две частицы находятся в точках с координатами 2 и 4, соответственно. В момент времени 2 они сталкиваются в точке 3 и исчезают. Больше в БЛК частиц нет.

# Задача B. Счет в гипершашках

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

Андрей работает судьей на чемпионате по гипершашкам. В каждой игре в гипершашки участвует три игрока. По ходу игры каждый из игроков набирает некоторое положительное целое число баллов. Если после окончания игры первый игрок набрал $a$ баллов, второй — $b$, а третий $c$, то говорят, что игра закончилась со счетом $a$:$b$:$c$.

Андрей знает, что правила игры гипершашек устроены таким образом, что в результате игры баллы любых двух игроков различаются не более чем в $k$ раз. После матча Андрей показывает его результат, размещая три карточки с очками игроков на специальном табло. Для этого у него есть набор из $n$ карточек, на которых написаны числа $x_1$, $x_2$, …, $x_n$. Чтобы выяснить, насколько он готов к чемпионату, Андрей хочет понять, сколько различных вариантов счета он сможет показать на табло, используя имеющиеся карточки.

Требуется написать программу, которая по числу $k$ и значениям чисел на карточках, которые имеются у Андрея, определяет количество различных вариантов счета, которые Андрей может показать на табло.

## Формат входных данных

Первая строка входного файла содержит два целых числа: $n$ и $k$ ($3 \leqslant n \leqslant 100\,000$, $1 \leqslant k \leqslant 10^9$).

Вторая строка входного файла содержит $n$ целых чисел $x_1$, $x_2$, …, $x_n$ ($1 \leqslant x_i \leqslant 10^9$).

## Формат выходных данных

Выходной файл должен содержать одно целое число — искомое количество различных вариантов счета.

## Примеры

| game.in | game.out |
|---|---|
| 5 2<br>1 1 2 2 3 | 9 |

## Замечания

В приведенном примере Андрей сможет показать следующие варианты счета: 1:1:2, 1:2:1, 2:1:1, 1:2:2, 2:1:2, 2:2:1, 2:2:3, 2:3:2, 3:2:2. Другие тройки чисел, которые можно составить с использованием имеющихся карточек, не удовлетворяют заданному условию, что баллы любых двух игроков различаются не более чем в $k = 2$ раза.

# Задача C. Странные строки

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

Рассмотрим строку $s$, состоящую из строчных букв латинского алфавита. Примером такой строки является, например, строка «`abba`».

Подстрокой строки $s$ называется строка, составленная из одного или нескольких подряд идущих символов строки $s$. Обозначим как $W(s)$ множество, состоящее из всех возможных подстрок строки $s$. При этом каждая подстрока входит в это множество не более одного раза, даже если она встречается в строке s несколько раз. Например, $W(\text{«}\texttt{abba}\text{»}) = \{\text{«}\texttt{a}\text{»}, \text{«}\texttt{b}\text{»}, \text{«}\texttt{ab}\text{»}, \text{«}\texttt{ba}\text{»}, \text{«}\texttt{bb}\text{»}, \text{«}\texttt{abb}\text{»}, \text{«}\texttt{bba}\text{»}, \text{«}\texttt{abba}\text{»}\}$.

Подпоследовательностью строки $s$ называется строка, которую можно получить из $s$ удалением произвольного числа символов. Обозначим как $Y(s)$ множество, состоящее из всех возможных подпоследовательностей строки $s$. Аналогично $W(s)$, каждая подпоследовательность строки $s$ включается в $Y(s)$ ровно один раз, даже если она может быть получена несколькими способами удаления символов из строки $s$. Поскольку любая подстрока строки $s$ является также ее подпоследовательностью, то множество $Y(s)$ включает в себя $W(s)$, но может содержать также и другие строки. Например, $Y(\text{«}\texttt{abba}\text{»}) = W(\text{«}\texttt{abba}\text{»}) \cup \{\text{«}\texttt{aa}\text{»}, \text{«}\texttt{aba}\text{»}\}$. Знак $\cup$ обозначает объединение множеств.

Будем называть строку $s$ *странной*, если для нее $W(s) = Y(s)$. Так, строка «`abba`» не является странной, а, например, строка «`abb`» является, так как для нее $W(\text{«}\texttt{abb}\text{»}) = Y(\text{«}\texttt{abb}\text{»}) = \{\text{«}\texttt{a}\text{»}, \text{«}\texttt{b}\text{»}, \text{«}\texttt{ab}\text{»}, \text{«}\texttt{bb}\text{»}, \text{«}\texttt{abb}\text{»}\}$.

Будем называть *странностью строки* число ее различных странных подстрок. При вычислении странности подстрока считается один раз, даже если она встречается в строке $s$ в качестве подстроки несколько раз. Так, для строки «`abba`» ее странность равна 7, любая ее подстрока, кроме всей строки, является странной.

Требуется написать программу, которая по заданной строке $s$ определяет ее странность.

## Формат входных данных

Входной файл содержит строку $s$, состоящую из строчных букв латинского алфавита. Строка имеет длину от 1 до 200 000.

## Формат выходных данных

Выходной файл должен содержать одно целое число: странность заданной во входном файле строки.

## Примеры

| strange.in | strange.out |
|---|---|
| abba | 7 |