Яндекс. Тренировки по алгоритмам июнь 2021, занятие 5

G. Счет в гипершашках

Ограничение времени	1 секунда
Ограничение памяти	64Mb
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

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 $x_1, x_2, \ldots, x_n$. 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 ввода

Первая строка входного файла содержит два целых числа: n и k ($3 \le n \le 100000$, $1 \le k \le 10^9$).
Вторая строка входного файла содержит n целых чисел $x_1, x_2, \ldots, x_n$ ($1 \le x_i \le 10^9$).

Формат вывода

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

Пример

Ввод	Вывод
5 2 1 1 2 2 3	9

Примечания

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.

Язык Python 3.12.1

Набрать здесь | Отправить файл

```python
from collections import Counter

# считываем данные
n, k = map(int, input().split())
cards = list(map(int, input().split()))

# подсчет количества одинаковых карт
count = Counter(cards)

# сортировка уникальных карт
uniqnums = list(count.keys())
uniqnums.sort()

rg = 0
answer = 0
duplicates = 0
# проходимся левой границей раздвижного окна по uniqnums
for lf in range(len(uniqnums)):
    # пока левое меньше правого в не более k раз
    while rg < len(uniqnums) and uniqnums[lf] * k >= uniqnums[rg]:
        # подсчитываем дубликаты
        if count[uniqnums[rg]] >= 2:
            duplicates += 1
        rg += 1
    # длина текущего раздвижного окна
    window_len = rg - lf
    # добавляем 3 варианта ответа на каждое оставшееся число при счетчике >= 2
    if count[uniqnums[lf]] >= 2:
        answer += (window_len - 1) * 3
    # добавляем 1 вариант ответа при счетчике >= 3
    if count[uniqnums[lf]] >= 3:
        answer += 1
    # добавляем варианты ответа для прогрессии в текущем раздвижном окне, используя его длину
    answer += (window_len - 1) * (window_len - 2) * 3

    # убираем дубликаты, посчитанные ранее с uniqnums[lf]
    if count[uniqnums[lf]] >= 2:
```

Отправить

Предыдущая | Следующая