

## Разбиение

Темур и его помощник Улугбек готовят фокус для шоу *Узбекистан ищет таланты*. Суть фокуса в том, что Темур решает **задачу о разбиении**: разбить заданный набор положительных целых чисел на  $K$  непустых групп так, чтобы сумма элементов в каждой группе была одной и той же. Другими словами, каждое число из набора должно быть назначено ровно одной из  $K$  групп, и сумма чисел в каждой группе должна быть одинаковой. Например, если задан набор  $[2, 1, 6, 4, 5]$  и  $K = 3$ , то допустимым разбиением будет  $[2, 4]$ ,  $[1, 5]$  и  $[6]$ . В этом случае сумма чисел в каждой группе равна 6.

Фокус выполняется следующим образом:

- Улугбек и Темур заходят в отдельные кабинки и не могут общаться друг с другом.
- Член жюри передаёт Улугбеку массив  $A$  из  $N$  положительных целых чисел  $A[0], A[1], \dots, A[N-1]$ , где каждое значение лежит в диапазоне от 1 до  $M$  включительно. Член жюри также передаёт Улугбеку значение  $K$ .
- Улугбек выбирает не более  $K - 1$  (не обязательно различных) целых чисел, чтобы **добавить их в качестве новых элементов** в массив  $A$ . Каждое добавленное число также должно лежать в диапазоне от 1 до  $M$  включительно.
- Жюри добавляет к исходному массиву числа, выбранные Улугбеком. Затем этот расширенный массив **сортируется в неубывающем порядке** и передаётся Темуру вместе со значением  $K$ .
- Затем Темур должен решить задачу о разбиении для этого расширенного отсортированного массива.

Ваша задача — разработать и реализовать стратегию для Темура и Улугбека. Можно доказать, что при заданных ограничениях существует стратегия, позволяющая им успешно решить задачу разбиения независимо от массива  $A$ , предоставленного жюри.

## Детали реализации

Вы должны реализовать две функции.

Функция, которую надо реализовать для **Улугбека**, выглядит следующим образом:

```
std::vector<int> add_numbers(std::vector<int> A, int K, int M)
```

- $A$ : массив длиной  $N$  — исходный массив, переданный Улугбеку.

- $K$ : целевое количество групп; обратите внимание, что Улугбек может добавить в массив до  $K - 1$  чисел.
- $M$ : максимально допустимое значение для каждого исходного и вновь добавленного числа.
- Данная функция вызывается ровно один раз для каждого теста.

Функция должна возвращать массив  $C$ , содержащий числа, которые Улугбек хочет добавить к исходному массиву. Пусть  $S$  обозначает длину массива  $C$ .

- $S$  не должно превышать  $K - 1$ .
- Каждое значение в  $C$  должно находиться в диапазоне от 1 до  $M$  включительно.

Функция, которую надо реализовать для **Темура**, выглядит следующим образом:

```
std::vector<int> find_partition(std::vector<int> B, int K)
```

- $B$ : массив длиной  $N + S$ , содержащий как исходные числа из  $A$ , так и числа, добавленные Улугбеком. Элементы массива  $B$  отсортированы в неубывающем порядке.
- $K$ : целевое количество групп.
- Данная функция вызывается ровно один раз для каждого теста.

Функция должна возвращать массив  $P$ , представляющий разбиение  $B$  на  $K$  непересекающихся групп.

- Длина массива  $P$  должна составлять  $N + S$ .
- Для каждого  $j$  ( $0 \leq j < N + S$ )  $P[j]$  обозначает группу, к которой принадлежит  $B[j]$ .
- Группы должны быть пронумерованы от 0 до  $K - 1$ , то есть для каждого  $0 \leq j < N + S$  должно выполняться  $0 \leq P[j] < K$ .
- Для каждого  $i$  от 0 до  $K - 1$  включительно должен существовать по крайней мере один элемент  $j$  ( $0 \leq j < N + S$ ), такой что  $P[j] = i$ .
- Сумма элементов, назначенных каждой из  $K$  групп, должна быть одинаковой.

При оценке программа, вызывающая вышеуказанные функции, запускается **ровно два раза**.

- Во время первого запуска программы:
  - `add_numbers` вызывается ровно один раз.
  - Возвращённый массив обрабатывается тестирующей системой для вычисления массива  $B$ .
- Во время второго запуска программы:
  - `find_partition` вызывается ровно один раз.

## Ограничения

- $3 \leq N \leq 100\,000$
- $2 \leq K \leq 100\,000$
- $K \leq N$
- $1 \leq M \leq 10^9$
- $1 \leq A[i] \leq M$  для каждого  $i$  ( $0 \leq i < N$ ).

## Подзадачи

| Подзадача | Баллы | Дополнительные ограничения      |
|-----------|-------|---------------------------------|
| 1         | 5     | $N = 3$                         |
| 2         | 4     | $M = 1$                         |
| 3         | 7     | $M \leq 2$                      |
| 4         | 10    | $N \leq 10$                     |
| 5         | 7     | $K = 2$                         |
| 6         | 12    | $K \leq 3$                      |
| 7         | 17    | $K \leq 10$                     |
| 8         | 20    | $K \leq 100$                    |
| 9         | 18    | Нет дополнительных ограничений. |

## Пример

Рассмотрим вызов:

```
add_numbers([8, 2, 9, 6, 1, 5, 5], 3, 9)
```

В этом примере  $A = [8, 2, 9, 6, 1, 5, 5]$ . Мы хотим разбить числа на  $K = 3$  групп. Улугбек может добавлять числа в диапазоне от 1 до  $M = 9$ .

Функция может вернуть  $[5, 4]$  — Улугбек решает добавить  $S = 2$  новых чисел: 5 и 4. Эти числа являются допустимыми, поскольку оба находятся в диапазоне от 1 до  $M = 9$ .

Затем расширенный массив сортируется и передается Темуру с помощью вызова:

```
find_partition([1, 2, 4, 5, 5, 5, 6, 8, 9], 3)
```

Мы можем разделить числа из этого массива на три группы:  $[1, 5, 9]$ ,  $[2, 5, 8]$  и  $[4, 5, 6]$ . Обратите внимание, сумма каждой из этих групп равна 15. Функция должна вернуть массив  $[0, 1, 2, 0, 1, 2, 2, 1, 0]$ .

## Грейдер

Формат ввода:

```
N K M
A[0] A[1] ... A[N-1]
```

Формат вывода:

После вызова `add_numbers` грейдер:

- Вычисляет отсортированный массив  $B$ .
- Выводит массивы  $C$  и  $B$  в следующем формате, **за которым следует пустая строка**:

```
S
C[0] C[1] ... C[S-1]
B[0] B[1] ... B[N+S-1]
```

После вызова `find_partition` грейдер возвращает:

```
L
P[0] P[1] ... P[L-1]
```

Здесь  $L$  — длина массива  $P$ , возвращённого `find_partition`.