

Partisi

Temur dan asistennya, Ulug'bek, sedang mempersiapkan trik sulap untuk acara *Uzbekistan Got Talent*. Triknya berkisar pada Temur yang menyelesaikan **permasalahan partisi** berikut: diberikan sekumpulan bilangan bulat positif, partisikan bilangan-bilangan itu menjadi K kelompok tak kosong sedemikian sehingga hasil penjumlahan semua elemen dalam masing-masing kelompok sama. Dengan kata lain, setiap bilangan bulat dari kumpulan tersebut harus dimasukkan dalam tepat satu dari K kelompok, dan hasil penjumlahan semua elemen dalam masing-masing kelompok harus sama. Sebagai contoh, jika kumpulan yang diberikan adalah $[2, 1, 6, 4, 5]$ dan $K = 3$, partisi yang valid adalah $[2, 4]$, $[1, 5]$, dan $[6]$. Di sini, hasil penjumlahan elemen dalam masing-masing kelompok tepat 6.

Trik ini dilaksanakan sebagai berikut:

- Ulug'bek dan Temur memasuki bilik terpisah dan tidak dapat berkomunikasi satu sama lain.
- Seorang anggota juri memberikan Ulug'bek sebuah *array* A yang berisi N bilangan bulat positif, $A[0], A[1], \dots, A[N - 1]$, yang setiap elemennya bernilai antara 1 dan M (inklusif). Anggota juri tersebut juga memberikan Ulug'bek nilai K .
- Ulug'bek memilih paling banyak $K - 1$ bilangan bulat (yang tidak harus berbeda) untuk **ditambahkan sebagai elemen baru** ke dalam *array* A . Setiap bilangan bulat yang ditambahkan juga harus bernilai antara 1 dan M (inklusif).
- Juri menambahkan bilangan bulat yang dipilih oleh Ulug'bek ke *array* awal. *Array* yang telah ditambahi ini kemudian **diurutkan secara tidak menurun** dan diberikan kepada Temur, bersama dengan nilai K .
- Temur kemudian harus menyelesaikan permasalahan partisi untuk *array* yang telah ditambahi dan diurutkan ini.

Tugas Anda adalah merancang dan menerapkan strategi untuk Temur dan Ulug'bek. Dapat dibuktikan bahwa, di bawah batasan yang diberikan, terdapat strategi yang memungkinkan mereka menyelesaikan permasalahan partisi, apa pun *array* A yang diberikan juri.

Detail Implementasi

Anda harus mengimplementasikan dua prosedur.

Prosedur yang harus Anda implementasikan untuk **Ulug'bek** adalah:

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

- A : array sepanjang N yang menyatakan array awal yang diberikan kepada Ulug'bek.
- K : banyaknya kelompok yang diinginkan; perhatikan bahwa Ulug'bek dapat menambahkan paling banyak $K - 1$ bilangan bulat ke dalam array.
- M : nilai maksimum yang diperbolehkan untuk setiap bilangan bulat yang ada di awal maupun yang akan ditambahkan.
- Prosedur ini dipanggil tepat sekali untuk setiap kasus uji.

Prosedur ini harus mengembalikan array C yang berisi bilangan bulat yang ingin ditambahkan Ulug'bek ke array awal. Misalkan S adalah panjang array C .

- S harus tidak lebih dari $K - 1$.
- Setiap elemen C harus bernilai antara 1 dan M (inklusif).

Prosedur yang harus Anda implementasikan untuk **Temur** adalah:

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

- B : array sepanjang $N + S$ yang berisi bilangan bulat awal dari A dan bilangan bulat yang ditambahkan oleh Ulug'bek. Elemen-elemen array B diurutkan secara tidak menurun.
- K : banyaknya kelompok yang diinginkan.
- Prosedur ini dipanggil tepat sekali untuk setiap kasus uji.

Prosedur ini harus mengembalikan array P yang menyatakan partisi dari B menjadi K kelompok yang saling lepas.

- Panjang P haruslah $N + S$.
- Untuk setiap j sehingga $0 \leq j < N + S$, $P[j]$ menandakan kelompok tempat $B[j]$ dimasukkan.
- Kelompok-kelompok tersebut harus dinomori 0 hingga $K - 1$, yang berarti $0 \leq P[j] < K$ harus berlaku untuk setiap $0 \leq j < N + S$.
- Untuk setiap i antara 0 dan $K - 1$ inklusif, harus ada setidaknya satu elemen j ($0 \leq j < N + S$) sedemikian sehingga $P[j] = i$.
- Hasil penjumlahan semua elemen yang dimasukkan ke masing-masing dari K kelompok harus sama.

Dalam proses penilaian sesungguhnya, program yang memanggil prosedur-prosedur di atas dijalankan **tepat dua kali**.

- Dalam perjalanan pertama:
 - `add_numbers` dipanggil tepat sekali.
 - Array yang dikembalikan akan diproses oleh sistem penilaian untuk menghitung array B .
- Dalam perjalanan kedua:
 - `find_partition` dipanggil tepat sekali.

Batasan

- $3 \leq N \leq 100\,000$
- $2 \leq K \leq 100\,000$
- $K \leq N$
- $1 \leq M \leq 10^9$
- $1 \leq A[i] \leq M$ untuk setiap i sehingga $0 \leq i < N$.

Subsoal

Subsoal	Nilai	Batasan Tambahan
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	Tidak ada batasan tambahan.

Contoh

Perhatikan pemanggilan berikut:

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

Pada contoh ini, $A = [8, 2, 9, 6, 1, 5, 5]$. Kita ingin memartisi bilangan-bilangan tersebut menjadi $K = 3$ kelompok. Ulug'bek dapat menambahkan bilangan antara 1 dan $M = 9$.

Prosedur ini dapat mengembalikan $[5, 4]$, yang berarti Ulug'bek memutuskan untuk menambahkan $S = 2$ bilangan bulat baru: 5 dan 4. Hal ini valid karena kedua elemen berada di antara 1 dan $M = 9$.

Array yang telah ditambahi kemudian diurutkan dan diberikan kepada Temur dengan pemanggilan berikut:

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

Kita dapat membagi bilangan bulat dari *array* ini menjadi tiga kelompok berikut: $[1, 5, 9]$, $[2, 5, 8]$, dan $[4, 5, 6]$. Perhatikan bahwa hasil penjumlahan setiap elemen dari masing-masing kelompok tersebut adalah 15. Untuk melaporkan partisi ini, prosedur tersebut harus mengembalikan *array* $[0, 1, 2, 0, 1, 2, 2, 1, 0]$.

Contoh *Grader*

Format masukan:

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

Format keluaran:

Setelah pemanggilan ke `add_numbers`, contoh *grader* akan:

- Menghitung *array* terurut B .
- Mengeluarkan *array* C dan B dalam format berikut, **diikuti satu baris kosong**:

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

Setelah pemanggilan `find_partition`, contoh *grader* akan mengeluarkan:

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

Di sini, L adalah panjang *array* P yang dikembalikan `find_partition`.