

Razdelitev

Temur in njegov pomočnik Ulug'bek pripravljata čarovniški trik za oddajo *Uzbekistan ima talent*. Za trik morata rešiti naslednji **problem razdelitve**: razdeliti dano polje naravnih števil v K nepraznih skupin, tako da je vsota elementov v vsaki skupini enaka. Z drugimi besedami, vsako število polja mora biti dodeljeno natanko eni izmed K skupin, pri čemer morajo biti vsote znotraj skupin enake. Na primer, če je podano polje $[2, 1, 6, 4, 5]$ in $K = 3$, je veljavna razdelitev $[2, 4]$, $[1, 5]$ in $[6]$. V tem primeru je vsota elementov v vsaki skupini natanko 6.

Trik izvedeta takole:

- Ulug'bek in Temur vstopita v ločeni kabini in med seboj ne smeta komunicirati.
- Na začetku sodnik Ulug'beku poda polje N naravnih števil A , $A[0], A[1], \dots, A[N - 1]$, kjer so vrednosti med vključno 1 in vključno M , ter K .
- Ulug'bek izbere največ $K - 1$ naravnih števil, ki jih želi **dodati kot nove elemente** polju A . Vsako dodano naravno število mora biti med vključno 1 in vključno M .
- Sodnik priloži Ulug'bekova števila prvotnemu polju. Tako razširjeno polje **uredi po nepadajočem vrstnem redu** in ga preda Temurju skupaj z vrednostjo K .
- Nato mora Temur to razširjeno urejeno polje razdeliti v K skupin.

Vaša naloga je zasnovati in implementirati strategijo za Temurja in Ulug'beka. Dokazati je mogoče, da pod danimi omejitvami obstaja strategija, ki jima omogoča uspešno razdelitev v K skupin ne glede na A .

Podrobnosti implementacije

Implementirati morate dve funkciji.

Funkcija, ki jo implementirate za **Ulug'beka**, je:

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

- A : polje dolžine N , ki predstavlja izvirno polje, podano Ulug'beku.
- K : ciljno število skupin. Ulug'bek lahko polju doda največ $K - 1$ naravnih števil.
- M : največja dovoljena vrednost vsakega izvirnega in novo dodanega naravnega števila.
- To funkcijo se kliče natanko enkrat za vsak testni primer.

Funkcija mora vrniti polje C , ki vsebuje naravna števila, ki želi Ulug'bek dodati izvirnemu polju. Naj bo S dolžina polja C .

- S je lahko največ $K - 1$.
- Vsak element C mora biti med vključno 1 in vključno M .

Funkcija, ki jo implementirate za **Temurja**, je:

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

- B : polje dolžine $N + S$, ki vsebuje tako izvorna naravna števila A kot naravna števila, ki jih je dodal Ulug'bek. Elementi B so razvrščeni v nepadajočem vrstnem redu.
- K : ciljno število skupin.
- To funkcijo se kliče natanko enkrat za vsak testni primer.

Funkcija mora vrniti polje P , ki predstavlja razdelitev B v K disjunktnih skupin.

- Dolžina P mora biti $N + S$.
- Za vsak j , kjer $0 \leq j < N + S$, $P[j]$ označuje skupino, ki pripada $B[j]$.
- Skupine morajo biti oštevilčene od 0 do $K - 1$, kar pomeni, da mora za vsak $0 \leq j < N + S$ veljati $0 \leq P[j] < K$.
- Za vsak i med vključno 0 in vključno $K - 1$ mora obstajati vsaj en element j ($0 \leq j < N + S$), da velja $P[j] = i$.
- Vsota elementov vsake izmed K skupin mora biti enaka.

Pri dejanskem ocenjevanju se program, ki kliče zgoraj omenjeni funkciji, zažene **natanko dvakrat**.

- Ob prvem zagonu programa:
 - `add_numbers` se pokliče natanko enkrat.
 - Ocenjevalni sistem uporabi vrnjeno polje za izračun polja B .
- Ob drugem zagonu programa:
 - `find_partition` se pokliče natanko enkrat.

Omejitve

- $3 \leq N \leq 100\,000$
- $2 \leq K \leq 100\,000$
- $K \leq N$
- $1 \leq M \leq 10^9$
- $1 \leq A[i] \leq M$ za vsak i , kjer $0 \leq i < N$.

Podnaloge

Podnaloga	Točke	Dodatne omejitve
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	Brez dodatnih omejitev.

Primer

Razmislite o naslednjem klicu:

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

V tem primeru je $A = [8, 2, 9, 6, 1, 5, 5]$. Števila želimo razdeliti v $K = 3$ skupine. Ulug'bek lahko doda števila med 1 in $M = 9$.

Funkcija lahko vrne polje $[5, 4]$, kar pomeni, da se Ulug'bek odloči dodati $S = 2$ novi števili 5 in 4. Ti števili sta veljavni, saj sta med 1 in $M = 9$.

Razširjeno polje se nato uredi in preda Temurju z naslednjim klicem:

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

Števila tega polja lahko razdelimo v naslednje tri skupine: $[1, 5, 9]$, $[2, 5, 8]$ in $[4, 5, 6]$. Opazimo lahko, da je vsota posameznih skupin enaka 15. Za poročanje o tej razdelitvi funkcija vrne polje $[0, 1, 2, 0, 1, 2, 2, 1, 0]$.

Vzorčni ocenjevalnik

Oblika vhoda

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

Oblika izhoda

Ko se klic `add_numbers` zaključi, vzorčni ocenjevalnik:

- izračuna urejeno polje B .
- izpiše polji C in B v naslednji obliki:

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

Ko se klic `find_partition` zaključi, vzorčni ocenjevalnik izpiše:

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

Tu je L dolžina polja P .