Fődíj

A játékban n dobozban (0-tól $n-1$-ig sorszámozva balról jobbra) vannak díjak. A díjak értékét csak a doboz kinyitása után lehet látni.

v ($v \geq 2$) különböző típusú díj van, 1-től v-ig sorszámozzuk, értékük szerint csökkenően. Azaz az 1-es a fődíj (gyémánt), amiből csak egy van. A legolcsóbb a v sorszámú díj, a nyalóka. Minden t típusra ($2 \leq t \leq v$) tudjuk, hogy ha k darab $t-1$ típusú díj van, akkor k^2-nél több darab t típusú díj van.

A cél a gyémánt megtalálása.

Kérdéseket tehetsz fel. Minden kérdésben egy dobozt jelölhetsz meg (i), válaszként egy kételemű tömböt kapsz, melynek jelentése a következő:

- Az $a[0]$ az i-től balra levő azon dobozok száma, amelyekben értékesebb ajándék van, mint i-ben.
- Az $a[1]$ az i-től jobbra levő azon dobozok száma, amelyekben értékesebb ajándék van, mint i-ben.

Például $n = 8$ és a kérdésben $i = 2$. A válasz $a = [1, 2]$ a következőt jelenti.

- A 0 és 1 sorszámú dobozok közül pontosan egy tartalmaz értékesebb ajándékot, mint a 2 doboz.
- A $3, 4, \ldots, 7$ sorszámú dobozok közül pontosan kettő tartalmaz értékesebb ajándékot, mint a 2 doboz.

írj programot, amely kevés kérdéssel megtalálja a gyémántot!

Megvalósítás

Az alábbi függvényt írd meg

```
int find_best(int n)
```

- n: a dobozok száma.
- A függvény értéke a gyémántot tartalmazó doboz d sorszáma legyen ($0 \leq d \leq n-1$), ahol a d. doboz az 1 tipusú ajándékot tartalmazza!

A megoldásodban ezt a függvényt hívhatod:

```
int[] ask(int i)
```

- i: annak a doboznak a sorszáma, amelyre rákérdezel ($0 \leq i \leq n-1$).
- Ez a függvény az a 2 elmű tömböt adja vissza. Az $a[0]$ az i-től balra levő azon dobozok száma, amelyekben értékesebb ajándék van, mint i-ben. Az $a[1]$ az i-től jobbra levő azon dobozok száma, amelyekben értékesebb ajándék van, mint i-ben.

Példa

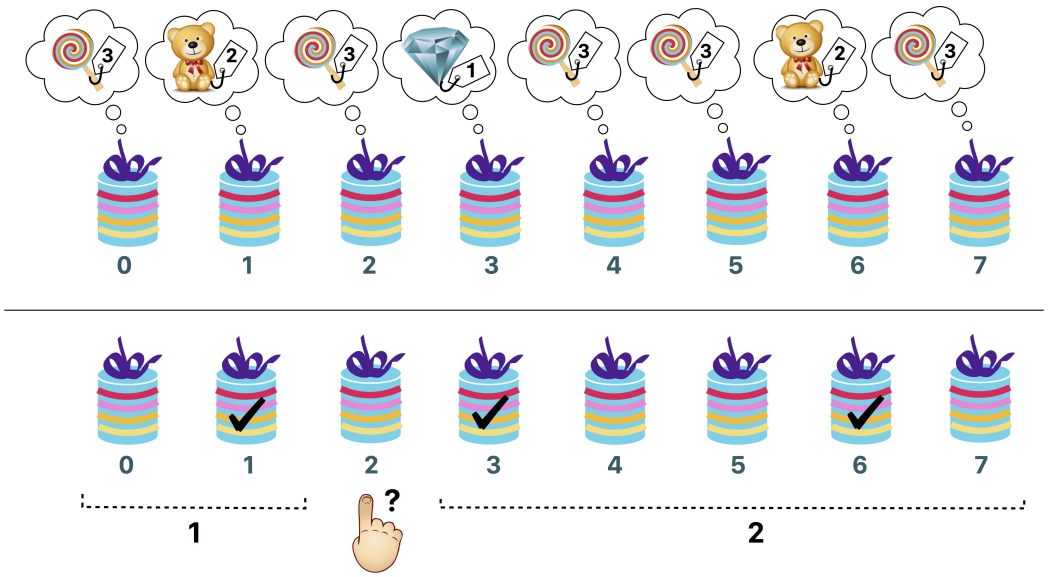
Az értékelő így hívja a függvényedet:

```
find_best(8)
```

$n = 8$ doboz van. Tegyük fel, hogy az ajándékok típusai: $[3, 2, 3, 1, 3, 3, 2, 3]$. Az `ask` lehetséges hívásai ezt adják eredményül.

- `ask(0)` returns $[0, 3]$
- `ask(1)` returns $[0, 1]$
- `ask(2)` returns $[1, 2]$
- `ask(3)` returns $[0, 0]$
- `ask(4)` returns $[2, 1]$
- `ask(5)` returns $[2, 1]$
- `ask(6)` returns $[1, 0]$
- `ask(7)` returns $[3, 0]$

A példában a gyémánt a 3. dobozban van, azaz a `find_best` eredménye 3 legyen!



Korlátok

- $3 \leq n \leq 200\,000$.
- Az ajándékok típusa 1 és v közötti egész szám.
- Pontosan egy 1-es típusú ajándék van.
- Minden t típusra ($2 \leq t \leq v$) tudjuk, hogy ha k darab $t-1$ típusú díj van, akkor k^2-nél több darab t típusú díj van.

Részfeladatok és pontozás

Néhány teszteset értékelésénél az értékelő adaptívan válaszol, amely azt jelenti, hogy az ajándékok sorozata nem előre rögzített. A válasz függhet attól, amit eddig kérdeztél. Biztosan teljesül, hogy a válasz olyan, hogy legalább egy olyan ajándék sorozat van, ami megfelel az eddigi kérdéseidre adott válaszokkal (csalfa válaszadó).

1. (20 pont) Pontosan 1 gyémánt és $n-1$ nyalóka van (tehát $v=2$). Az `ask` függvényt legfeljebb $10\,000$-szer hívhatod.
2. (80 pont) Nincs további korlátozás.

A 2 részfeladat pontozása a következő. Legyen q az `ask` függvény hívásai maximális száma a részfeladat összes tesztesetére! A pontokat az alábbiak szerint kapod:

Kérdések	Pontszám
$10\,000 < q$	0 (CMS-ben 'Wrong Answer' értékelés)
$6000 < q \leq 10\,000$	70
$5000 < q \leq 6000$	$80 - (q - 5000)/100$
$q \leq 5000$	80

Minta értékelő

A minta értékelő nem csalfa. Beolvas egy p tömböt az ajándék típusokkal. Minden b-re ($0 \leq b \leq n-1$) a b. dobozban levő ajándék típusa $p[b]$.

Bemenet formátuma:

- line 1: n
- line 2: $p[0]$ $p[1]$ $\ldots$ $p[n-1]$

Egyetlen sorba a `find_best` függvény értékét és az `ask` hívásai számát írja ki.