The Big Prize

Jocul "Marele premiu" este un renumit show de televiziune. Sunteți norocosul participant care s-a calificat în runda finală. Stați în fața unui rând de n cutii, numerotate de la stânga la dreapta de la 0 până la $n-1$. Fiecare cutie conține un premiu care nu poate fi văzut până când cutia nu este deschisă. Există $v \geq 2$ diferite *tipuri* de premii. Tipurile de premii sunt numerotate de la 1 la v în ordinea *descrescătoare* a valorii.

Premiul de tipul 1 este cel mai scump: un diamant. Există un singur diamant în cutii. Premiul de tip v este cel mai ieftin: o acadea. Pentru a face jocul cât mai captivant, numărul premiilor ieftine este mult mai mare decât numărul premiilor scumpe. Mai exact, pentru toate t cu $2 \leq t \leq v$ se știe: dacă există k premii de tip $t-1$, atunci există *strict mai mult* decât k^2 premii de tip t.

Aveți scopul să câștigați diamantul. La sfârșitul jocului va trebui să deschideți cutia și să luați premiul din cutie. Înainte de a deschide cutia aleasă trebuie să îl întrebați pe Rambod, gazda show-ului, câteva întrebări. Pentru fiecare întrebare, veți alege o cutie i. Ca răspuns, Rambod vă va oferi un șir a care conține doi întregi. Semnificația acestor întregi este:

- Printre cutiile din stânga cutiei i sunt exact $a[0]$ cutii care conțin premii mai scumpe ca premiul din cutia i.
- Printre cutiile din dreapta cutiei i sunt exact $a[1]$ cutii care conțin premii mai scumpe ca premiul din cutia i.

De exemplu, presupunem că $n = 8$ și alegeti cutia $i = 2$ ca întrebare pusă. Ca răspuns Rambod vă dă șirul $a = [1, 2]$. Semnificația acestui răspuns este:

- Exact una din cutiile 0 și 1 conține un premiu mai scump ca premiul din cutia 2.
- Exact două din cutiile $3, 4, \ldots, 7$ conțin premii mai scumpe ca premiul din cutia 2.

Sarcina voastră este să găsiți cutia care conține diamantul punând un număr mic de întrebări.

Detalii de implementare

Trebuie să implementați următoarea procedură:

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

- Această procedură este apelată exact o singură data de către evaluator.
- n: numărul de cutii.
- Această procedură returnează numărul cutiei care conține diamantul, adică un întreg unic d (

$0 \leq d \leq n-1$) astfel încât cutia d conține premiul de tip 1.

Procedura de mai sus poate apela următoarea procedură:

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

- i: numărul cutiei despre care vreți să întrebați. Valoarea lui i trebuie să fie între 0 și $n-1$, inclusiv.
- Această procedură returnează un șir a cu 2 elemente. $a[0]$ este numărul premiilor mai scumpe din cutiile din stânga cutiei i și $a[1]$ este numărul premiilor mai scumpe din cutiile din dreapta cutiei i.

Exemplu

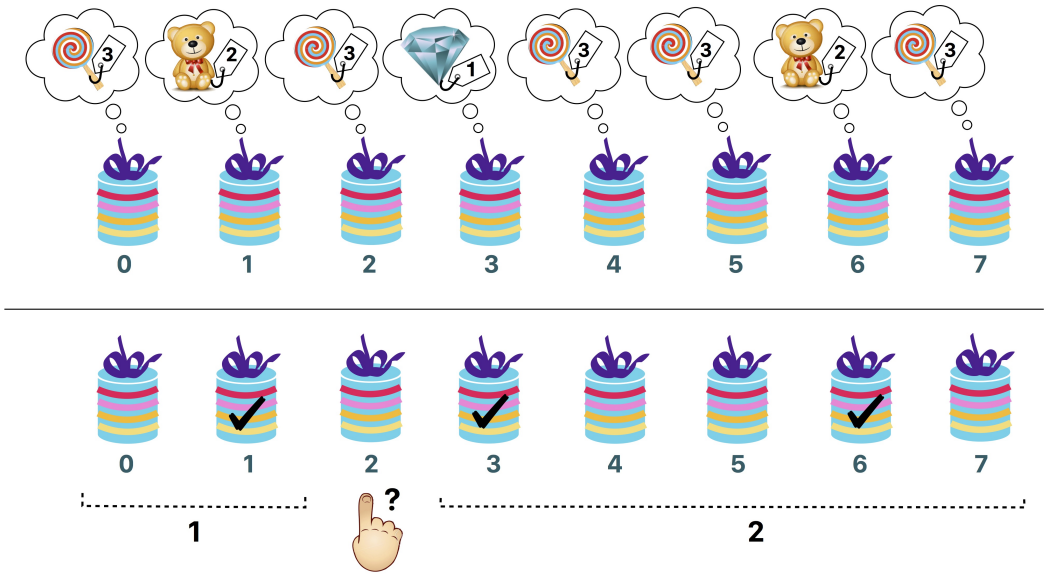
Evaluatorul execută următorul apel al procedurii:

```
find_best(8)
```

Există $n = 8$ cutii. Presupunem că premiile sunt de următoarele tipuri $[3, 2, 3, 1, 3, 3, 2, 3]$. Toate cazurile posibile de apel a procedurii `ask` și respectiv valorile returnate sunt:

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

În acest exemplu, diamantul este în cutia 3. Deci, procedura `find_best` trebuie să returneze 3.



Exemplul este ilustrat în figura de mai sus. Partea de sus arată valorile premiilor în fiecare cutie. Partea de jos arată interogarea `ask(2)`. Cutiile marcate conțin premii mai scumpe decât cutia cu numărul 2.

Restricții și precizări

- $3 \leq n \leq 200\,000$.
- Tipul premiilor în fiecare cutie este între 1 și v, inclusiv.
- Există un singur premiu de tipul 1.
- Pentru toate $2 \leq t \leq v$, dacă există k premii de tipul $t-1$, atunci există strict mai mult de k^2 premii de tipul t.

Subtask-uri și punctaj

În unele teste comportamentul evaluatorului este adaptiv. Adică, în aceste teste, evaluatorul nu are o secvență fixă de premii. În schimb, răspunsurile oferite de evaluator poate depinde de întrebările transmise de soluția voastră. Se garantează că evaluatorul răspunde în așa fel încât după fiecare răspuns există cel puțin o secvență consistentă de premii cu toate răspunsurile date până acum.

1. (20 puncte) Există exact 1 diamant și $n-1$ acadele (prin urmare, $v=2$). Puteți apela procedura `ask` de cel mult $10\,000$ ori.

2. (80 puncte) Fără restricții adiționale.

În subtask-ul 2 puteți obține un scor parțial. Fie q numărul maxim de apeluri a procedurii `ask` pentru toate testele din acest subtask. Punctajul pentru acest subtask este calculat conform următorului tabel:

Întrebări	Punctaj
$10\,000 < q$	0 (raportat în CMS ca 'Wrong Answer')
$6000 < q \leq 10\,000$	70
$5000 < q \leq 6000$	$80 - (q - 5000)/100$
$q \leq 5000$	80

Evaluator local

Evaluatorul local nu este adaptiv. În schimb, el citește și utilizează un șir fix p de tipuri de premii. Pentru toate $0 \leq b \leq n - 1$, tipul premiului din cutia b este dat ca $p[b]$. Formatul intrării pentru evaluatorul local este:

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

Evaluatorul local afișează o singură linie care conține valoarea returnată de `find_best` și numărul de apeluri a procedurii `ask`.