Голямата награда

"Голямата награда" е известна телевизионна игра. Вие сте играчът, който е класиран на финалния кръг. Пред вас е редица от n кутии. Отляво-надясно кутиите са номерирани с целите числа от 0 до $n-1$. Всяка кутия съдържа награда, която не може да я видите при затворена кутия. Наградите са $v \geq 2$ различни *вида*. Тези видове са номерирани от 1 до v по *намаляващ* ред на стойностите на наградите.

Награда от вид 1 е с най-голяма стойност: диамант. Има точно една награда от този вид. Наградата от вид v е с най-малка стойност: близалка. Броят на по-евтините награди е значително по-голям от броя на по-скъпите награди. По-точно, за всяко t, такова че $2 \leq t \leq v$ е изпълнено: ако съществуват k награди от вида $t-1$, тогава съществуват *строго повече* от k^2 награди от вида t.

Целта ви е да спечелите диаманта. В края на играта вие ще отворите една кутия и ще получитв наградата, която се съдържа там. Преди да изберете коя кутия да отворите, вие може да задавате въпроси на домакина на играта Rambod. При всеки от тези въпроси вие задавате номер на някоя кутия i. Като отговор Rambod ще ви даде един масив `a`, съдържащ две цели числа. Техният смисъл е следният:

- Измежду кутиите, които са наляво от кутия i, съществуват точно $a[0]$ кутии, всяка от които съдържа награда, която е по-скъпа от наградата в кутия i.
- Измежду кутиите, които са надясно от кутия i, съществуват точно $a[1]$ кутии, всяка от които съдържа награда, която е по-скъпа от наградата в кутия i.

Например нека $n = 8$. За ваш въпрос избирате номер на кутия $i = 2$. За отговор получавате масив `a = [1, 2]`. Смисълът е:

- Точно една от кутиите с номера 0 и 1 съдържа награда, която е по-скъпа от наградата, намираща се в кутия 2.
- Точно две от кутиите с номера $3, 4, \ldots, 7$ съдържат награда, която е по-скъпа от наградата, намираща се в кутия 2

Вашата задача е да намерите кутията, съдържаща диаманта чрез задаване на колкото се може по-малък брой въпроси.

Имплементация

Трябва да напишете следната процедура:

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

- Тази процедура се извиква от грейдера точно веднъж
- n: брой на кутиите
- Процедурата трябва да върне номера на кутията, която съдържа диаманта, т.е. едно цяло число d, $0 \leq d \leq n-1$ такова, че кутията с номер d съдържа награда от вид 1.

Гореописаната процедура може да прави извиквания на следната процедура:

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

- i: номер на кутия, за която вие задавате въпрос. Стойността на i трябва да бъде между 0 и $n-1$, включително.
- Процедурата връща масив с два елемента. Стойността на $a[0]$ е равна на броя на кутиите с по-скъпа награда, намиращите се отляво на кутията с номер i. Стойността на $a[1]$ е равна на броя на кутиите с по-скъпа награда, намиращите се отдясно на кутията с номер i.
- За означението `int[]` вижте първата таблица в Notice

Пример

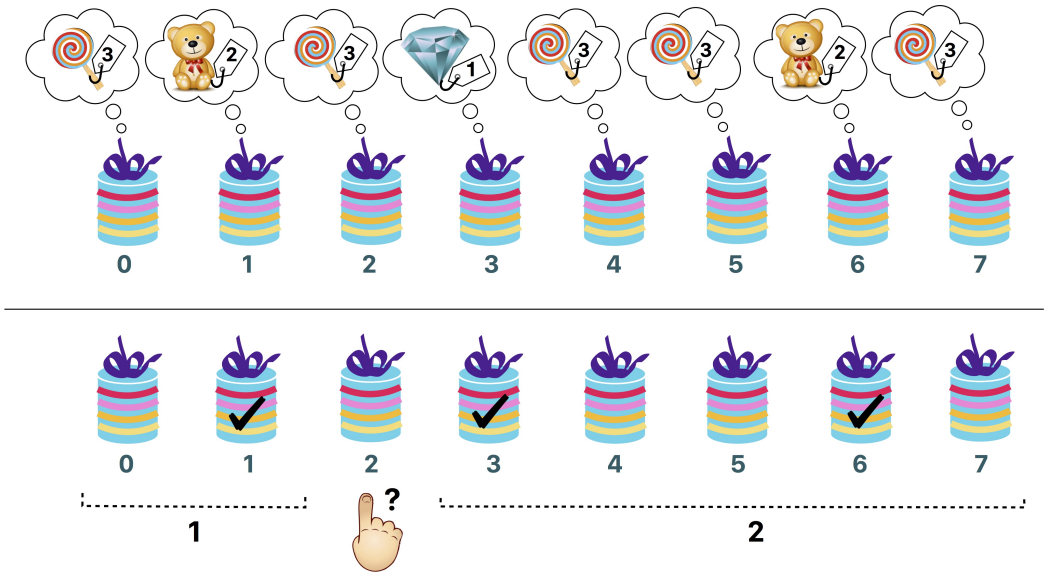
Грейдерът извика

```
find_best(8)
```

Това означава, че има $n = 8$ кутии. Предполагаме, че вида на наградите, разпредени в кутиите е $[3, 2, 3, 1, 3, 3, 2, 3]$. Всички възможни извиквания на процедурата `ask` и съответните върнати стойности от нея са:

- `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 $[3, 0]$

В този пример, диамантът е в кутия 3. Така процедурата find_best трябва да върне 3.



В горната част на фигурата са показаните стойностите на наградите във всяка кутия. Долната част илюстрира извикването `ask(2)`. Маркираните кутии съдържат по-скъпи награди от наградата в кутия 2.

Ограничения

- $3 \leq n \leq 200\,000$.
- Видът на наградата във всяка от кутиите е цяло число от 1 до v, включително.
- Съществува точно една награда от вид 1.
- За всяко t, $2 \leq t \leq v$, ако съществуват k награди от вид $t-1$, тогава съществуват *строго* повече от k^2 награди от вид t.

Подзадачи и оценяване

Поведението на грейдера е адаптивно при някои тестове. Това означава, че при тези тестове грейдерът няма фиксирана последователност на видовете награди. Тогава отговорите на грейдера могат да зависят от задаваните му въпроси. Гарантирано е, че отговорите на грейдера са такива, че след всеки отговор съществува поне една редица от видове награди, която е консистентна (съвместима) с всичките отговори, дадени от грейдера до момента.

1. (20 точки) Съществуват точно 1 диамант и $n-1$ близалки (т.е. $v = 2$). Може да извиквате процедурата `ask` най-много $10\,000$ пъти.
2. (80 точки) Няма допълнителни ограничения.

В подзадача 2 вие може да получите частично оценяване. Нека q е максималният брой извиквания на процедурата `ask` измежду всички тестове за тази подзадача. Тогава вашите точки за тази подзадача се пресмятат съгласно следната таблица:

Въпроси	Точки
$10\,000 < q$	0 (съобщавани от CMS като 'Wrong Answer')
$6000 < q \leq 10\,000$	70
$5000 < q \leq 6000$	$80 - (q - 5000)/100$
$q \leq 5000$	80

Примерен грейдер

Примерният грейдер не е адаптивен. Той просто чете фиксирана редица, стойностите в която са p вида награди и използва тази редица при даване на отговорите. За всяко b, $0 \leq b \leq n-1$, вида на наградата в кутия b се дава от грейдера като $p[b]$. Примерният грейдер очаква вход в следния формат:

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

Примерният грейдер отпечатва едининствен ред, съдържащ стойността, която връща `find_best` и броя на извикванията на процедурата `ask`.