Wielka nagroda

Wielka nagroda, to znany teleturniej. Jesteś szczęśliwym zawodnikiem, który dotarł do finału. Stoisz i patrzysz na n pudełek ustawionych w rzędzie i numerowanych od lewej do prawej liczbami od 0 do $n-1$. Każde pudełko zawiera nagrodę nieznanej wartości, któą poznać dopiero po otwarciu pudełka. Mamy $v \geq 2$ różnych *typów* nagród. Typy te są numerowane od 1 do v w *malejącym* porządku wartości.

Nagroda typu 1 jest najdroższa: diament. Diament jest dokładnie w jednym pudełku. Nagroda typu v jest najtańsza: lizak. Aby grę uczynić bardziej emocjonującą, zdecydowano że tanich nagród będzie znacznie więcej niż drogich. Dokładniej: dla każdego t takiego że $2 \leq t \leq v$ wiemy, że: jeśli nagród o wartości $t-1$ jest k, to nagród o wartości t jest więcej niż k^2. Twoje zadanie, to znaleźć diament. Na końcu gry będziesz mógł otworzyć wskazane przez Ciebie pudełko i otrzymać nagrodę, którą zawiera. Zanim wskażesz pudełko, możesz zadać kilka pytań Rambodowi -- prowadzącemu program. Każde pytanie polega na wybraniu jakiegoś pudełka o numerze i. Rambod przekaże Ci tablicę a z dwiema wartościami całkowitymi. Ich znaczenie jest następujące:

- Pośród pudełek na lewo od pudełka i mamy dokładnie $a[0]$ pudełek z bardziej wartościowymi nagrodami, niż ta w pudełku i.
- Pośród pudełek na prawo od pudełka i mamy dokładnie $a[1]$ pudełek z bardziej wartościowymi nagrodami, niż ta w pudełku i.

Dla przykładu: załóżmy, że $n = 8$. Jako pytanie wybierasz pudełko o numerze $i = 2$. Rambod odpowiada, że $a = [1, 2]$. Jego odpowiedź oznacza, że:

- Dokładnie jedno pudełko spośród 0 i 1 zawiera nagrodę większą niż nagroda w pudełku 2.
- Dokładnie dwa pudełka spośród pudełek $3, 4, \ldots, 7$ zawierają nagrodę większą niż nagroda w pudełku 2. Twoje zadanie polega na znalezieniu diamentu za pomocą możliwie małej liczby zapytań.

Szczegóły implementacyjne

Powinieneś zaprogramować następującą funkcję:

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

- Funkcja ta jest wywoływana dokładnie raz przez sprawdzaczkę.
- n: liczba pudełek.
- Funkcja ta powinna wyznaczyć etykietę pudełka zawierającego diament, czyli jedyną liczbę całkowitą d z przedziału $0 \leq d \leq n-1$ taką, że pudełko d zawiera nagrodę typu 1.

Funkcja ta może wywoływać następującą funkcję:

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

- i: numer pudełka, o które chcesz zapytać. Wartość i powinna być z przedziału od 0 do $n-1$ włącznie.
- funkcja `ask` zwraca tablicę a z 2 wartościami. Wartość $a[0]$ to liczba bardziej wartościowych nagród w pudełkach na lewo od pudełka i, zaś $a[1]$ to liczba bardziej wartościowych nagród w pudełkach na prawo od i.

Przykład

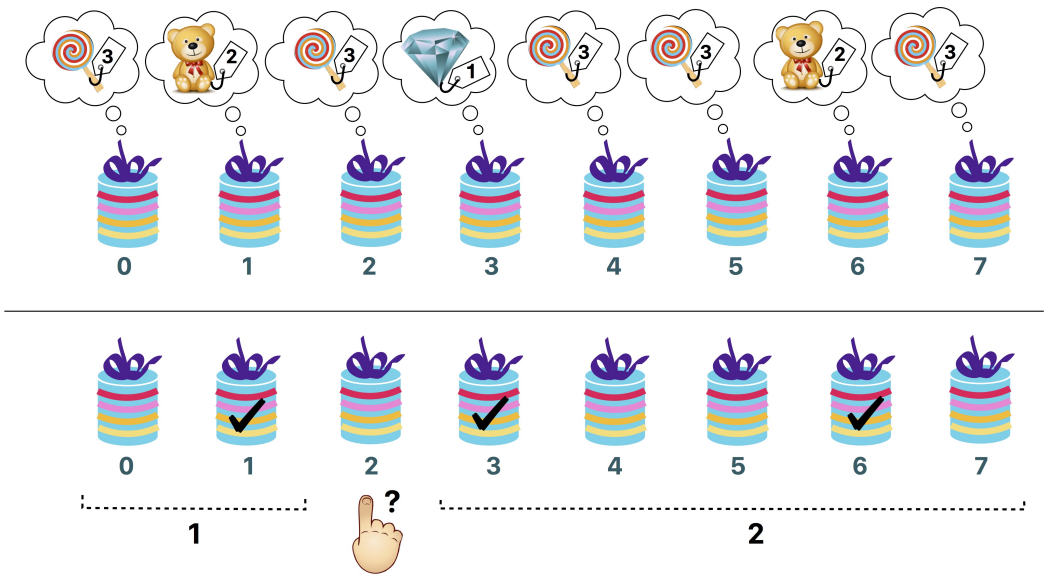
Sprawdzaczka wywołuje następującą funkcję:

```
find_best(8)
```

Mamy $n = 8$ pudełek. Załóżmy, że typy nagród w tych pudełkach to $[3, 2, 3, 1, 3, 3, 2, 3]$. Poniżej przedstawiono wszystkie możliwe wywołania funkcji `ask` wraz z odpowiedziami.

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

Diament w tym przykładzie znajduje się w pudełku 3, więc funkcja `find_best` powinna zwrócić 3.



Powyższy rysunek ilustruje podany przykład. Na górze widzimy typy nagród w każdym pudełku. W dolnej części widzimy efekt wywołania `ask(2)`. Zaznaczone pudełka zawierają większe nagrody niż ta z pudełka 2.

Ograniczenia

- $3 \leq n \leq 200\,000$.
- Typ nagrody w każdym pudełku mieści się w przedziale od 1 do v włącznie.
- Jest dokładnie jedno pudełko z nagrodą typu 1.
- Dla każdego $2 \leq t \leq v$, jeśli k nagród jest typu $t-1$, to *ściśle* więcej niż k^2 nagród jest typu t.

Podzadania i ocenianie

W niektórych zadaniach zachowanie sprawdzaczki jest adaptacyjne. Oznacza to, że w takich przypadkach sprawdzaczka nie ma z góry ustalonej sekwencji nagród. Odpowiedzi generowane przez sprawdzaczkę mogą zależeć od pytań zadawanych przez Twoją funkcję. Masz zagwarantowane, że sprawdzaczka odpowiada w taki sposób, że zawsze co najmniej jedna sekwencja nagród jest zgodna ze wszystkimi dotychczas udzielonymi odpowiedziami.

1. (20 punktów) Dokładnie jeden diament i $n-1$ lizaków (tutaj, $v = 2$). Funkcja `ask` może być wywołana co najwyżej $10\,000$ razy.
2. (80 punktów) Brak dodatkowych ograniczeń.

W podzadaniu 2 można uzyskać częściowy wynik. Niech q będzie największą liczbą wywołań funkcji `ask` pośród wszystkich testów dla tego podzadania. Wtedy wynik Twojego podzadania obliczany jest według następującego schematu:

Pytania	Wynik
$10\,000 < q$	0 (zgłaszane przez CMS jako 'Wrong Answer')
$6000 < q \leq 10\,000$	70
$5000 < q \leq 6000$	$80 - (q - 5000)/100$
$q \leq 5000$	80

Przykładowa sprawdzaczka

Przykładowa sprawdzaczka nie jest adaptacyjna. Po prostu wczytuje i wykorzystuje ustaloną tablicę p typów nagród. Dla każdego $0 \leq b \leq n-1$ typ nagrody w pudełku b wynosi $p[b]$. Przykładowa sprawdzaczka spodziewa się wejścia w następującym formacie:

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

Przykładowa sprawdzaczka wypisuje pojedynczy wiersz zawierający wynik wywołania funkcji `find_best` oraz liczbę wywołań funkcji `ask`.