Большой приз

Большой приз — известная телевикторина. Вам повезло стать одним из участников финального раунда. Напротив вас в ряд выставлены n коробок. Коробки пронумерованы целыми числами от 0 до $n-1$ слева направо. В каждой из коробок находится приз, который невозможно увидеть, не открыв коробку. В телевикторине используются призы v различных *типов*, где $v \geq 2$. Типы пронумерованы целыми числами от 1 до v в порядке *убывания* ценности призов.

Самый дорогой приз — бриллиант (тип 1). Ровно в одной коробке находится бриллиант. Самый дешевый приз — леденец (тип v). Чтобы игра была более захватывающей, количество более дешевых призов значительно больше, чем количество более дорогих. Строго говоря, для всех t ($2 \leq t \leq v$) известно следующее: если имеется k призов типа $t-1$, то призов типа t *строго больше* k^2.

В конце игры вы должны открыть коробку и получить приз, который в ней находится. Ваша задача состоит в том, чтобы выиграть бриллиант. До выбора коробки, которая будет открыта, вы можете задать ведущему телевикторины Рамбоду несколько вопросов. Для каждого вопроса вы выбираете некоторую коробку с номером i. В ответ Рамбод предоставит массив a, состоящий из двух целых чисел. Эти числа означают следующее:

- Среди всех коробок, расположенных слева от коробки с номером i, ровно $a[0]$ коробок содержат приз дороже, чем приз в коробке с номером i.
- Среди всех коробок, расположенных справа от коробки с номером i, ровно $a[1]$ коробок содержат приз дороже, чем приз в коробке с номером i.

Например, предположим, что $n = 8$. Для вопроса вы выбрали коробку с номером $i = 2$, и в ответ Рамбод говорит, что $a = [1, 2]$. Этот ответ означает следующее:

- Ровно одна коробка из коробок с номерами 0 и 1 содержит приз дороже, чем коробка с номером 2.
- Ровно две коробки из коробок с номерами $3, 4, \ldots, 7$ содержат призы дороже, чем коробка с номером 2.

Требуется найти коробку, содержащую бриллиант, уложившись в требуемое количество вопросов.

Детали реализации

Вам следует реализовать следующую функцию (метод):

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

- Эта функция вызывается из проверяющего модуля ровно один раз.
- n: количество коробок.
- Функция должна вернуть номер коробки, в которой находится бриллиант, то есть такое уникальное целое число d ($0 \leq d \leq n - 1$), что в коробке с номером d содержится приз типа 1.

В этой функции разрешено вызывать следующую функцию:

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

- i: номер коробки, которую вы выбрали для вопроса. Значение i должно быть от 0 до $n - 1$ включительно.
- Функция возвращает массив a из двух элементов. Значение $a[0]$ равно количеству более дорогих призов, расположенных в коробках слева от коробки с номером i. Значение $a[1]$ равно количеству более дорогих призов, расположенных в коробках справа от коробки с номером i.

Пример

Проверяющий модуль делает следующий вызов функции:

```
find_best(8)
```

Используются $n = 8$ коробок со следующими типами призов $[3, 2, 3, 1, 3, 3, 2, 3]$. Все возможные вызовы функции `ask` и возвращаемые значения перечислены ниже.

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

В этом примере бриллиант находится в коробке с номером 3. Возвращаемое значение функции `find_best` должно быть равно 3.

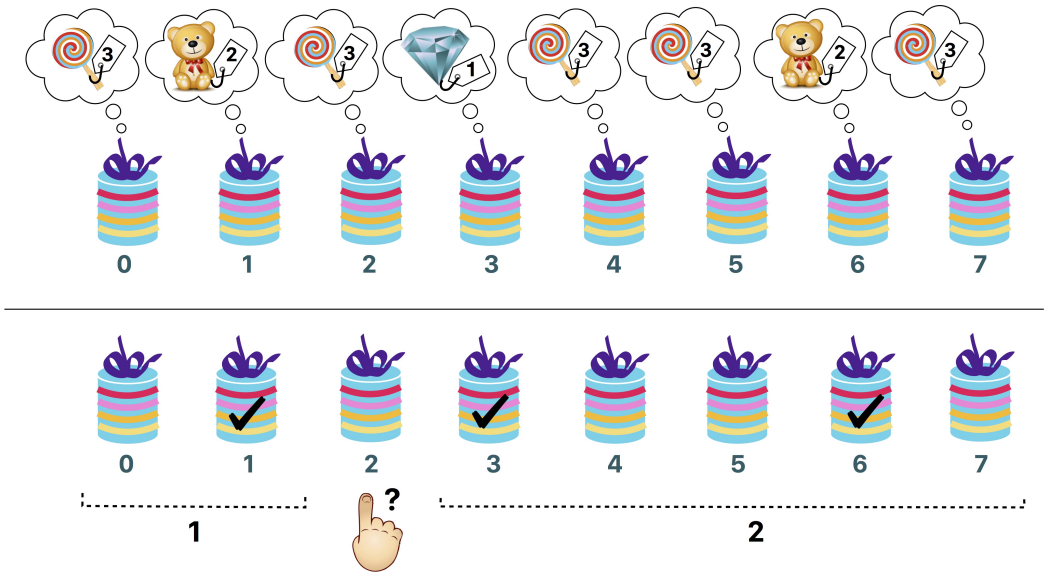

Рисунок выше иллюстрирует пример. Верхняя часть показывает значения типов призов в каждой из коробок. Нижняя часть иллюстрирует вызов `ask(2)`, отмеченные коробки содержат более дорогие, чем приз из коробки с номером 2.

Ограничения

- $3 \leq n \leq 200\,000$.
- Типы призов в коробках лежат в диапазоне от 1 до v включительно.
- Ровно одна коробка содержит приз типа 1.
- Для всех $2 \leq t \leq v$, если используется k призов типа $t-1$, то призов типа t используется *строго* больше k^2.

Система оценивания

В некоторых случаях поведение проверяющего модуля адаптивно. Это значит, что проверяющий модуль не использует фиксированной последовательности призов. Вместо этого ответы от проверяющего модуля могут зависеть от запросов, которые делает ваша программа. Гарантируется, что проверяющий модуль отвечает таким образом, что после каждого ответа существует не менее одной последовательности призов, соответствующей всем ответам, данным до этого.

1. (20 баллов) Среди призов один бриллиант и $n-1$ леденец (то есть $v = 2$). Вы можете вызвать функцию `ask` не более $10\,000$ раз.
2. (80 баллов) Без дополнительных ограничений.

В подзадаче 2 вы можете получить часть баллов. Пусть q равно максимальному количеству вызовов функции `ask` среди всех тестов в подзадаче. Тогда оценка в этой подзадаче вычисляется в соответствии со следующей таблицей:

Количество вызовов функции	Результат
$10\,000 < q$	0 (вердикт в CMS будет 'Wrong Answer')
$6000 < q \leq 10\,000$	70
$5000 < q \leq 6000$	$80 - (q - 5000)/100$
$q \leq 5000$	80

Пример проверяющего модуля

Пример проверяющего модуля не является адаптивным. Вместо этого он читает и использует фиксированный массив p типов призов. Для всех $0 \leq b \leq n - 1$ тип приза в коробке с номером b задается величиной $p[b]$.

Пример проверяющего модуля читает входные данные в следующем формате:

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

Пример проверяющего модуля печатает отдельную строку, содержащую возвращаемое функцией `find_best` значение и количество вызовов функции `ask`.