커다란 상품

커다란 상품은 유명한 TV 게임 쇼이다. 당신은 다행히도 결승전까지 진출했다. 당신 앞에는 n 개의 상자가 한 줄로 놓여져 있는데, 왼쪽부터 차례로 0부터 $n-1$까지 번호가 매겨져 있다. 각각의 상자에는 상자를 열 때까지는 볼 수 없는 상품 하나가 들어 있다. 상은 서로 다른 $v \geq 2$ **종류**가 있다. 이 종류는 가치의 **내림차순**으로 1부터 v까지 번호가 매겨져 있다.

1번 종류 상품은 가장 비싼 것으로 다이아몬드이다. 다이아몬드가 들어있는 상자는 정확히 하나이다. v번 종류 상품은 가장 싼 것으로 사탕이다. 게임을 더 재미있게 하기 위해서, 싼 종류의 상품의 수는 이보다 비싼 종류의 상품 수보다 훨씬 많다. 보다 구체적으로, $2 \leq t \leq v$인 t에 대해서 다음 조건이 성립한다. 만약 $t-1$ 종류의 상품이 k개 있다면, t 종류 상품은 k^2 개보다 **많이** 있다.

당신의 목적은 다이아몬드를 얻는 것이다. 게임이 모두 끝날 때 당신은 상자 하나를 열어야 하고, 이 상자에 들어있는 상품을 받는다. 열 상자를 정하기 전에, 당신은 게임의 사회자인 Rambod에게 질문들을 할 수 있다. 각각의 질문으로, 당신은 어떤 상자 i를 지정한다. Rambod는 질문에 대한 답으로, 두 정수가 저장된 배열 a를 준다. 이것의 의미는 다음과 같다.

- 상자 i의 왼쪽에 있는 상자들 중에서 정확히 $a[0]$개의 상자에는 상자 i에 들어있는 상품보다 비싼 상품이 들어 있다.
- 상자 i의 오른쪽에 있는 상자들 중에서 정확히 $a[1]$개의 상자에는 상자 i에 들어있는 상품보다 비싼 상품이 들어 있다.

예를 들어, $n = 8$이라고 하자. 질문으로, 당신이 $i = 2$번 상자를 지정했다. 이에 대한 답으로 Rambod는 $a = [1, 2]$라고 대답했다. 이 답의 의미는 다음과 같다.

- 0번 상자와 1번 상자 중 정확히 하나에 2번 상자에 들어있는 상품보다 비싼 상품이 들어있다.
- $3, 4, \ldots, 7$번 상자 중 정확히 둘에 2번 상자에 들어있는 상품보다 비싼 상품이 들어있다.

당신의 임무는 적은 수의 질문을 통해서 다이아몬드가 있는 상자를 찾는 것이다.

Implementation details

다음 함수를 구현해야 한다.

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

- 이 함수는 그레이더에 의해 단 한번 호출된다.
- n: 상자의 수.
- 이 함수의 리턴값은 다이아몬드가 들어있는 상자의 번호이다. 즉, 리턴값이 정수 d ($0 \leq d \leq n-1$) 라면 상자 d에 1번 종류 상품이 들어있다.

위 함수는 다음 함수를 호출할 수 있다.

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

- i: 당신이 질문하는 상자의 번호. i 값은 0 이상 $n-1$ 이하이다.
- 이 함수의 리턴값은 원소가 2개인 배열 a이다. $a[0]$ 은 상자 i 왼쪽에 있는 상자들에 들어있는 더 비싼 상품의 수이며, $a[1]$은 상자 i 오른쪽에 있는 상자들에 들어있는 더 비싼 상품의 수이다.

Example

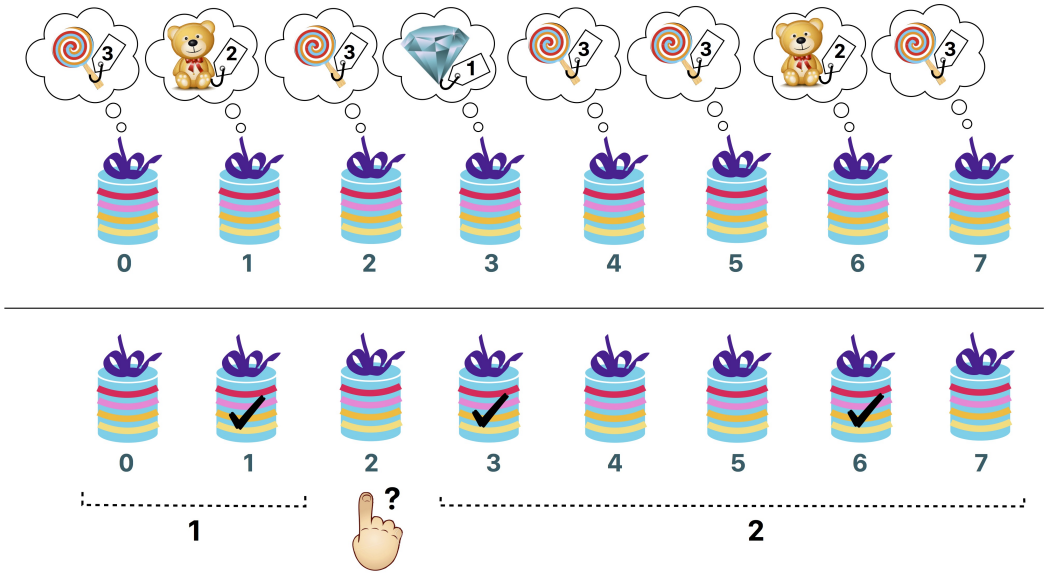
그레이더는 다음 함수를 호출한다.

```
find_best(8)
```

$n = 8$ 개의 상자가 있다. 상품들이 상자에 차례로 $[3, 2, 3, 1, 3, 3, 2, 3]$의 형태로 주어졌다고 하자. 모든 가능한 `ask` 함수 호출과 그에 해당하는 함수의 리턴값은 다음과 같다.

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

이 예제에서, 다이아몬드는 상자 3에 있다. 따라서 함수 `find_best`의 리턴값은 3이어야 한다.



위 그림이 예제를 설명하고 있다. 위쪽은 각 상자에 들어있는 상품의 가치를 보여주고 있고, 아래쪽은 질의 `ask(2)`에 대한 답을 보여주고 있다. V자로 표시된 상자에는 2번 상자에 들어 있는 상품보다 비싼 상

품이 들어 있다.

Constraints

- $3 \le n \le 200\,000$.
- 각각의 상자에 들어 있는 상품의 종류는 1 이상 v 이하.
- 1번 종류의 상품은 정확하게 하나 있다. .
- 모든 $2 \le t \le v$에 대해서, $t-1$ 종류의 상품이 k개 있다면, t 종류의 상품은 k^2 개 **초과**로 있다.

Subtasks and scoring

어떤 테스트케이스에서는 그레이더의 동작이 적응적(adaptive)이다. 즉, 이 테스트케이스에서는 그레이더는 처음 시작할 때 상자 안에 든 상품의 순서를 고정해놓지 않는다. 당신이 하는 질문들에 따라 해당하는 질의의 답은 달라질 수 있다. 그레이더가 매번 답변을 할 때마다, 지금까지 한 답변들과 일치하는 상품들의 서열이 적어도 하나 이상 존재하는 것은 보장된다.

1. (20 points) 정확히 1개의 다이아몬드와 $n-1$ 개의 사탕이 있다. (즉, $v = 2$) `ask` 함수를 최대 $10\,000$ 번 호출할 수 있다.
2. (80 points) 추가적인 제약조건이 없다.

서브태스크 2에서는 부분 점수가 있다. q가 이 서브태스크에 속한 모든 테스트케이스 중에서 `ask`를 가장 많이 호출한 경우의 호출 횟수라고 하자. 그렇다면, 이 서브태스크에서 얻는 점수는 다음 식과 같다.

질의의 수	점수
$10\,000 < q$	0 (CMS의 판정은 'Wrong Answer')
$6000 < q \le 10\,000$	70
$5000 < q \le 6000$	$80 - (q - 5000)/100$
$q \le 5000$	80

Sample grader

샘플 그레이더는 적응적이지 않다. 대신, 상품의 종류가 저장된 고정된 배열 p를 읽은 뒤 사용한다. 모든 $0 \le b \le n-1$에 대해 상자 b에 들어있는 상품의 종류는 $p[b]$이다. 샘플 그레이더는 다음 포맷으로 입력이 들어온다고 가정한다.

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

샘플 그레이더는 의 `find_best`의 리턴값과, `ask`의 호출 횟수를 한 줄에 출력한다.