

# Magic Trick

Alicia and Beatriz are preparing a magic trick for the IOI Talent Show. The trick works as follows:

- A volunteer selects a permutation  $P$  of length  $N$  and places  $N$  cards on the table. The cards are numbered from 0 to  $N - 1$ , with card  $i$  displaying the value  $P[i]$ .
- Alicia enters the room, observes the cards, and selects  $K$  of them to flip face down, hiding their values.
- Beatriz then enters the room, sees the current arrangement of the cards (including which ones are face down), and magically determines the values of all  $K$  hidden cards!

Your task is to devise and implement a strategy for Alicia and Beatriz. The more impressive the trick, the better your score: the objective is to maximize  $K$ , the number of hidden cards Beatriz can correctly reveal.

## Implementation Details

The **first** procedure that you have to implement:

```
std::vector<int> Alicia(std::vector<int> P)
```

- $P$ : array of length  $N$ , representing the selected permutation.

This procedure should return an array  $Q$  of length  $N$ , representing the card flips Alicia performs. For each index  $i$  ( $0 \leq i \leq N - 1$ ), the values in  $Q$  must be set as follows:

- $Q[i] = -1$  if Alicia flips card  $i$ , and
- $Q[i] = P[i]$  otherwise.

The **second** procedure that you have to implement:

```
std::vector<int> Beatriz(std::vector<int> Q)
```

- $Q$ : array of length  $N$ , as returned by `Alicia`. This array specifies the configuration of the cards when Beatriz enters.

This procedure should return an array  $B$  of length  $N$ , representing the original permutation  $P$ , that is,  $B[i] = P[i]$  should hold for each  $i$  ( $0 \leq i \leq N - 1$ ).

In each test case, the two procedures are called exactly once, as follows:

- During the first run of the program:
  - `Alicia` is called with the original permutation  $P$ .
  - For the array  $Q$  returned by `Alicia`:
    - If the array does not conform to the constraints described above, you receive an `Output isn't correct verdict`.
    - Otherwise, array  $Q$  is stored by the grading system.
- During the second run of the program:
  - `Beatriz` is called with the array  $Q$ .

## Constraints

- $N = 256$
- $1 \leq P[i] \leq N$  for each  $i$  such that  $0 \leq i < N$ .
- All values in  $P$  are distinct.

## Scoring

Let  $M$  be the minimum value of  $K$  for which your solution successfully performs the trick across all test cases.

- If  $M = 0$  or your solution fails to correctly reconstruct the permutation  $P$  in any test case, you receive 0 points.
- Otherwise, your score is:

$$\min(20 + 5 \cdot M, 100)$$

In particular, a full score is achieved if  $M = 16$ .

## Example

Consider a scenario where  $N = 6$  and  $P = [2, 4, 3, 1, 5, 6]$ . The procedure `Alicia` is called as:

```
Alicia([2, 4, 3, 1, 5, 6])
```

Suppose `Alicia` uses the following strategy: flip every card  $i$  such that  $P[i] = i + 1$ . In this case, the condition holds for  $i = 2, 4$ , and  $5$ . Hence, the procedure returns the array  $[2, 4, -1, 1, -1, -1]$ .

Now, the procedure `Beatriz` is called as:

```
Beatriz([2, 4, -1, 1, -1, -1])
```

Knowing the strategy of `Alice`, she finds and returns the original array  $P = [2, 4, 3, 1, 5, 6]$ .

In this case,  $K = 3$ , as three cards were flipped. However, if submitted, this strategy would receive a score of 0, because there exist permutations where no index  $i$  satisfies  $P[i] = i + 1$ .

Note that this example does not satisfy the constraint  $N = 256$  and therefore will not be used during grading. The downloadable attachment for this task includes a sample input for the grader with  $N = 256$ . The same permutation  $P$  is used in Subtask 0 during evaluation.

## Sample Grader

Input format:

```
N
P[0] P[1] ... P[N-1]
```

Output format:

```
S
Q[0] Q[1] ... Q[S-1]
T
B[0] B[1] ... B[T-1]
```

Here:

- $S$  is the length of the array  $Q$  returned by `Alicia`.
- $T$  is the length of the array  $B$  returned by `Beatriz`.

Note that while  $N = 256$  holds for all testcases in this task, you may use the sample grader with any value of  $N$ .