찹쌀떡[1] (chaltteok)

찹쌀떡이 불티나게 팔리는 수능 시즌이 지나가고, 찹쌀떡 공장에도 드디어 평화가 찾아왔다! 찹쌀떡 공장의 공장장 지훈이는 N종류의 찹쌀떡을 한 개씩 일렬로 배치하여 포장하는 세트만을 판매하며, 모든 제품에서 찹쌀떡의 배치 순서는 동일하다. 각 종류의 찹쌀떡마다 1번, 2번, ..., N번의 번호를 붙였는데, 번호를 붙이는 순서는 찹쌀떡의 배치 순서와 다를 수 있다.

경쟁 회사의 찹쌀떡을 생산하는 공장의 공장장 수찬이는 지훈이네 찹쌀떡 공장에서 판매하는 세트의 찹쌀떡 배치 순서가 궁금했고, 공장이 놀고 있어서 심심했던 지훈이와 간단한 게임을 하기로 하였다. 게임은 여러 턴으로 구성되며, 한 턴은 아래와 같이 진행된다.

수찬이는 N개의 찹쌀떡 중 일부를 선택한다. 지훈이는 수찬이가 선택한 찹쌀떡만을 포장 세트의 원래 위치에 배치하고 연속한 찹쌀떡이 모두 붙어벼렸을 때의 찹쌀떡 덩어리의 개수를 수찬이에게 알려준다.

예를 들어, 찹쌀떡의 배치 순서가 4번, 2번, 3번, 1번, 5번 순서였고, 첫 번째 턴에서 수찬이가 2, 3, 5번 떡을 선택했을 때를 생각해보자. 이때 지훈이는 2번과 3번 떡이 서로 붙어서 하나의 덩어리를 형성하고 5번 떡이 다른 하나의 덩어리를 형성하므로, 첫 번째 턴의 답으로 2를 수찬이에게 알려준다.

턴이 무한히 진행되면 답을 항상 알 수 있으므로, 지훈이는 20 000턴이 진행되면 더 이상 대답해주지 않는다. 그러므로 수찬이는 20 000턴 이내로 세트 내의 찹쌀떡 배치 순서를 알고 싶어한다. 수찬이를 도와 찹쌀떡 세트의 배치 순서를 구하는 프로그램을 작성해보자.

입력 형식

첫 번째 줄에 찹쌀떡의 수 N ($1 \le N \le 1\,000$)이 주어진다.

상호 작용(Interaction)

턴을 시작하려면, '`?` M i_1 i_2 $\cdots$ i_M'를 한 개의 줄에 출력해야 한다. M은 수찬이가 선택한 떡의 개수를 나타내며, $i_1, i_2, \cdots, i_M$은 수찬이가 선택한 떡의 번호를 의미한다. 예를 들어, 문제의 설명과 같이 2, 3, 5번의 떡을 선택한다면, '`? 3 2 3 5`' 또는 '`? 3 3 2 5`'와 같이 한 개의 줄에 출력하면 된다. (떡의 번호를 출력하는 순서가 반드시 오름차순일 필요는 없다.) 같은 번호를 2번 이상 출력하거나, 출력한 번호가 1 이상 N 이하가 아니면 오답 판정을 받게 된다. 턴은 최대 20 000번 진행되고, 20 001번 이상 턴을 진행하는 경우 오답 판정을 받게 된다.

세트의 찹쌀떡 배치 순서를 알아냈다면, '`!` p_1 p_2 $\cdots$ p_N'를 한 개의 줄에 출력하고, 게임을 종료한다. 여기서 $p_1, p_2, \cdots, p_N$는 구한 찹쌀떡 배치 순서를 나타낸다. 찹쌀떡의 실제 배치 순서와 출력한 순서가 동일하다면, 정답 판정을 받는다. 예를 들어, 위의 예시가 주어졌을 때 '`! 4 2 3 1 5`'를 출력하면 정답 판정을 받는다. 문제의 특성상, 출력한 순서가 실제 배치 순서를 뒤집은 것과 같아도 정답으로 인정된다. 즉, '`! 5 1 3 2 4`'를 출력해도 정답 판정을 받는다. 그러나 이외의 경우, 예를 들어 '`! 1 2 3 4 5`'를 출력하면 오답 판정을 받는다.

매 줄을 출력할 때마다 C/C++에서는 '`fflush(stdout);`', C++에서는 '`std::cout << std::flush;`', Java에서는 '`System.out.flush();`'를 통해 표준 출력 버퍼를 비워줘야 한다.

여러분의 구현을 돕기 위해, 채점 시스템에서 C/C++/Java 예시 코드를 내려받을 수 있다. 이미 구현되어 있는 '`pack`' 함수와 '`report`' 함수를 그대로 사용하면 된다.

채점 방식

1. (17점) $N \le 200$, 항상 1번 찹쌀떡이 첫 번째로 배치된다.
2. (18점) $N \le 200$
3. (65점) 추가 제약 조건 없음.

[1] https://en.wikipedia.org/wiki/Chapssal-tteok

참고

문제 페이지 하단에서 내려받을 수 있는 `helper.zip`에는 아래 파일들이 들어 있다.

- `chaltteok.(c|cpp|java)`: 예시 코드이다. "`/* IMPLEMENT HERE */`" 사이의 부분을 구현하면 된다. 항상, '? N 1 2 ... N' 또는 '! 1 2 ... N'을 출력하며, 이 코드를 그대로 제출하면 점수를 받을 수 없다.
- `simulator.py`: 실제 채점 시스템처럼 프로그램과 입출력을 주고 받는 Python 스크립트이다. Windows 10, Ubuntu 18.04 (WSL), Mac OS X의 Python 2.7, 3.6에서 테스트되었으며, Python 버전이 다르면 실행이 제대로 되지 않을 수 있다.
  - 간단한 사용법: 명령줄에 '`python simulator.py -h`'를 입력하면 사용 가능한 옵션들을 볼 수 있다.
    - `--input INPUT`: 이 옵션을 사용하면 스크립트가 `INPUT` 파일에서 테스트 데이터를 읽는다. 입력 형식은 아래 '`example.in`' 부분을 참고한다.
    - `-l`, `--log`: 이 옵션을 켜면, 스크립트와 여러분이 작성한 프로그램 사이의 입출력 경과를 모두 출력한다.
    - `--limit LIMIT`: 턴 최대 진행 횟수 제한이다. 명시하지 않았을 때의 기본값은 문제에 제시한 것과 같은 20 000이다.
    - 모든 옵션을 입력한 뒤, 마지막에 코드를 실행하는 명령어를 입력해야 한다. 명령어를 따옴표로 감싸면 안 된다.
  - 예시:
    - `python simulator.py --limit 100 -l ./chaltteok`
    - `python simulator.py --log chaltteok.exe`
    - `python simulator.py --input input.txt java chaltteok`
- `example.in`: 아래의 형식으로 찹쌀떡 배치에 대한 정보를 스크립트에게 제공한다.
  - 1번째 줄: N
  - 2 - $(N+1)$번째 줄: i번째 줄에서 주어지는 수에 대응되는 찹쌀떡은, 찹쌀떡 세트에서 $i-1$번째에 배치된다.