# 붕어빵

**[문제]** $k$개의 붕어빵 틀(frame)이 일렬로 연결된 붕어빵 기계가 있다. 우리는 이 기계를 이용해서 붕어빵을 구워내려고 한다. 붕어빵은 앞 뒤 면이 서로 다르게 구별되도록 만들어져 있다. 이 붕어빵 틀은 배열(array) **Fish[k]**로 표현된다. 이 빵틀은 좀 독특하게 구성되어 있어서 틀의 일부 연속된 구간을 전체 단위로 "휙" 뒤집을 수 있다. 물론 1개만을 뒤집을 수도 있다.

예를 들어 특정 일부 구간 **Fish[i:j]**를 뒤집으면 j번째 항목이 i번에 오고 i번째 항목은 j번으로 이동한다. 즉 수평 방향으로 각 내용이 선대칭적(symmetric)으로 이동하는 것이다. 이러한 작업을 flipping이라고도 한다. 아래 그림을 이용해서 설명해보자. 그림은 12개의 틀로 이루어진 붕어빵 기계를 배열로 보여주고 있다. 1부터 12까지 표시된 번호는 개별 틀의 index를 의미한다. 붕어빵은 1부터 12까지의 정수로 표시되어 있다. 만일 번호 $k$인 붕어빵이 Flip으로 한번 뒤집어지면 이 숫자는 $-k$로 표시되고 다시 뒤집어지면 $-(-k)=k$로 원래의 양의 정수로 되돌아 온다. 초기 상태는 모든 숫자가 양수이며 순차적으로 들어가 있어 다음과 같이 표현된다.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |

**flip[4,9]:** 만일 이 상태에서 4번부터 9번까지를 뒤집으면 빵틀은 다음과 같이 변한다.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | -9 | -8 | -7 | -6 | -5 | -4 | 10 | 11 | 12 |

**flip[6,11]:** 다시 이 상태에서 6번부터 11번까지를 뒤집으면 다음과 같이 변한다.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | -9 | -8 | -11 | -10 | 4 | 5 | 6 | 7 | 12 |

**flip[2,4]:** 이 상태에서 구간 [2,4] 를 뒤집으면 다음과 같이 변한다.

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 9 | -3 | -2 | -8 | -11 | -10 | 4 | 5 | 6 | 7 | 12 |

이 flip 작업을 역순으로 전개하면 우리는 처음 상태의 붕어빵 구성(configuration)을 얻을 수 있다. 예를 들어 위 상태가 [1, 9, -3, -2, -8, -11, -10, 4, 5, 6, 7, 12]인 상태에서 **flip[2,4]** → **flip[6,11]** → **flip[4,9]** 수행하면 [1, 2, 3, .. 12]를 만들 수 있고 이 상태에서 붕어빵을 모두 꺼낸다. 여러분에게는 최종 붕어빵틀의 상태가 정수로 주어진다. 여러분은 이 상태를 보고 최

소한의 flip[] 작업을 이용하여 최초의 상태 [1, 2, 3, ... $k$]를 만들어야 한다. 즉 최종 입력 상태를 읽어서 원상 복구에 필요한 최소한의 flip 작업의 수를 계산해야 한다.

만일 1번의 작업으로 충분하다면 문자열 “one", 2번으로 가능하다면 문자열 ”two", 그리고 그 이상 3번 이상의 작업이 필요하다면 “over"를 정답으로 출력해야 한다. 단 이번 문제에서 초기 상태, 즉 한번도 뒤집지 않는 경우는 입력으로 들어오지 않는다. 따라서 답은 항상 one, two, over 중에 있다.

**[입출력]** 입력과 출력은 표준 입출력 파일인 **stdin과 stdout을** 사용한다. 입력 파일 stdin의 첫 줄에는 붕어빵틀의 크기 $k$가 장수로 주어진다. 각 test case마다 5개씩의 subcases가 준비되어 있다. 즉 이어지는 5개의 줄에는 붕어빵틀의 최종 상태를 나타내는 $k$개의 정수 순서가 주어진다. 각 정수는 하나 이상의 공백으로 분리되어 있다. 당연히 입력 정수는 $-k$부터 $k$범위의 서로 다른 정수로 구성되어 있다. 따라서 여러분은 {”one", "two", "over" } 중에서 정답을 골라 5개 subcase의 순서대로 한 줄에 하나씩 5개의 줄에 출력해야 한다. 이 문제에서 입력 데이터 $k$의 범위는 $5 \le k \le 10{,}000$ 이다.

**[예제]**

| stdin | stdout |
|---|---|
| 6<br>1 2 -4 -3 5 6<br>1 2 4 -3 5 6<br>-3 -2 6 1 -5 -4<br>4 5 -1 -6 2 3<br>6 1 -5 -4 2 3 | one<br>two<br>over<br>over<br>over |
| 12<br>1 2 3 4 5 6 7 8 9 -10 11 12<br>1 -2 3 4 5 6 7 8 9 -10 11 12<br>1 -2 3 -4 5 -6 7 -8 9 10 -11 12<br>1 9 -3 -2 -8 -11 -10 4 5 6 7 12<br>1 -12 -11 -10 7 8 9 -6 -5 -4 -3 -2 | one<br>two<br>over<br>over<br>two |

**[제한조건]** 프로그램의 이름은 pa03_fish.{py,c,cpp,java}이다. 제출 횟수는 최대 15번이며 허용 시간은 데이터 당 제한 시간은 1초, 허용가능 코드의 최대 크기는 3,000 bytes 이다. 문제 풀이 마감시간은 2022년 4월10일 24:00 이다. 제출한 프로그램에 대한 풀이(방법과 코드설명)를 작성하여 2022년 4월12일 24:00까지 NESPA “설명게시판”에 제출해야 한다. 제출한 프로그램 풀이과정은 마감이 지나면 공개된다.