문제 2. 도서관

입력 파일:	`standard input`
출력 파일:	`standard output`
시간 제한:	2초
메모리 제한:	256MB

수백 년의 시간이 지난 끝에, JOI 도시는 폐허가 되었다. 탐험가 IOI양은 도서관이 있었던 지역을 조사하고 있다. 조사를 통해 다음과 같은 것들을 알 수 있다.

- JOI 도시의 도서관 서재에는 N 권의 책이 있었다. N 권의 책은 책장에 왼쪽에서 오른쪽으로 일렬로 놓여 있었다.
- N 권의 책은 1번부터 N번까지의 번호가 붙어있다. 하지만 책장에 꽂힌 순서와 번호가 붙은 순서는 다를 수도 있다.
- 작업 한 번으로 책장에서 연속으로 놓인 책을 가져갈 수 있었다.

유감스럽게, IOI양은 도서관에서 오래된 책을 찾는 데에 실패했다. 하지만 도서관의 작업을 담당하는 기계를 찾았다. 한 개 이상의 책 번호를 기계한테 물어볼 경우, 기계는 이 책들을 가져가기 위해서 필요한 작업의 최소횟수를 답해주었다.

IOI양은 기계에 질문을 하면서 책이 위치한 순서를 알고 싶어한다. 단, N개의 책이 역순으로 놓여있는 경우에도 답은 변하지 않으므로 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다.

기계가 오래되었기 때문에 최대 20 000번의 질문만 할 수 있다.

구현 명세

당신은 파일 하나를 제출해야 한다.

이 파일의 이름은 `library.cpp`이다. 파일은 다음 함수를 구현해야 한다. 또한, `library.h`를 include해야 한다.

- `void Solve(int N)`

  이 함수는 각 테스트 케이스마다 정확히 한 번 불린다.

  - 인자 `N`은 책의 수 N을 나타낸다.

  당신의 프로그램은 다음 함수를 호출 할 수 있다.

  - `int Query(const std::vector<int>& M)`

    한 개 이상의 책 번호를 기계에 물어볼 경우, 이 함수는 책들을 가져가기 위해서 필요한 작업의 최소횟수를 반환한다.

    * 가져갈 책의 번호들은 인자 `M`으로 표시되며, 이는 크기 N의 'vector'이다. 각 i $(1 \leq i \leq N)$에 대해, `M[`i`-1]`$= 0$인 경우, i 번째 책은 가져가지 않는다. `M[`i`-1]`$= 1$인 경우, i 번째 책은 가져간다. 만약 `M`의 크기가 N과 다를 경우 **오답** **[1]**이 된다. 각 i에 대해, `M[`i`-1]`은 0 혹은 1이어야 한다. `M[`i`-1]`$= 1$인 i가 $(1 \leq i \leq N)$ 적어도 한 개는 존재해야 한다. 이 두 조건을 만족하지 않을 경우 **오답** **[2]**이 된다. `Query`를 20 000번 초과로 호출할 경우, **오답** **[3]**이 된다.

  - `void Answer(const std::vector<int& res)`

    이 함수를 이용하여 책꽂이에 꽂힌 책의 순서를 답할 수 있다. 책이 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다.

    * 인자 `res`는 크기 N의 'vector'이다. 이는 책꽂이에 꽂혀있는 책의 순서를 나타낸다. 각 i $(1 \leq i \leq N)$ 에 대해 책꽂이에 왼쪽에서 i 번째로 꽂혀있는 책의 번호는 `res[`i`-1]`이다. 만약 `res`의 크기가 N과 다를 경우, **오답** **[4]**이 된다. 만약 `res[`i`-1]`는 1 이상 N 이하의 수여야 한다. 만약 이를 만족하지 않을 경우, **오답** **[5]**이 된다. 또한, 수 `res[0]`, `res[1]`, `res[`N`-1]`은 모두 달라야 한다. 만약 이를 만족하지 않을 경우, **오답** **[6]**이 된다.

Solve함수가 종료될 때, Answer함수를 호출한 횟수가 한 번이 아니면, **오답** [7]이 된다. 만약 Solve로 주어진 책의 순서가 책장에 꽂힌 순서와 다르면 **오답** [8]이 된다. 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경쓰지 않는다.

참고 사항

- 당신의 프로그램은 내부에서 사용할 목적으로 함수나 전역변수를 사용할 수 있다.
- 당신의 프로그램은 표준 입출력을 사용해서는 안 된다. 당신의 프로그램은 어떠한 방법으로도 다른 파일에 접근해서는 안 된다. 단, 당신의 프로그램은 디버그 목적으로 표준 에러출력에 출력할 수 있다.

당신은 대회 홈페이지의 아카이브에서 프로그램을 테스트하기 위한 목적의 샘플 그레이더를 받을 수 있다. 아카이브는 당신의 프로그램의 예제 소스 또한 첨부되어 있다. 샘플 그레이더는 파일 grader.cpp이다. 당신의 프로그램을 테스트하기 위해서, grader.cpp, library.cpp, library.h를 같은 디렉토리 안에 놓고, 컴파일하기 위해 다음 커맨드를 실행하여라.

- `g++ -std=c++14 -O2 -o grader grader.cpp library.cpp`

컴파일이 성공적이면, 파일 grader가 생성된다.

실제 그레이더와 샘플 그레이더는 다름에 주의하여라. 샘플 그레이더는 하나의 프로세스에서 실행되며, 입력을 표준 입력으로부터 받고, 출력을 표준 출력에 출력한다.

입력 형식

샘플 그레이더는 표준 입력에서 다음과 같은 형식으로 입력받는다.

- 첫째 줄에는 정수 N이 주어진다. 이는 책꽂이에 책이 N 권 있다는 의미이다.
- 다음 N 개의 줄의 i 번째 ($1 \leq i \leq N$) 줄에는 정수 A_i가 주어진다. 이는 왼쪽에서 i 번째 꽂힌 책의 번호가 A_i임을 의미한다.

출력 형식

프로그램이 정상적으로 종료되었다면, 샘플 그레이더는 다음과 같은 정보를 표준 출력에 출력한다. (따옴표는 출력하지 않는다.)

- 정답으로 판단된 경우, Query함수의 호출 횟수를 "Accepted: 100"과 같은 형식으로 출력한다.
- 오답으로 판단된 경우, 오답의 종류를 "Wrong Answer [1]"과 같은 형식으로 출력한다.

프로그램이 다양한 오답의 종류에 속해 있으면 샘플 그레이더는 그중 하나만 출력할 것이다.

제한

모든 입력 데이터는 다음의 조건을 만족한다. N과 A_i의 의미는, 입력 형식을 참고하여라.

- $1 \leq N \leq 1\,000$.
- $1 \leq A_i \leq N$ $(1 \leq i \leq N)$.
- $1 \leq A_i \neq A_j \leq N$ $(1 \leq i < j \leq N)$.

서브태스크 1 (19 점)

- $N \leq 200$.

서브태스크 2 (81 점)

추가 제한조건이 없다.

예제

이 함수는 그레이더의 예제 입력과 해당하는 함수 호출을 보여준다.

예제 입력	예제 함수 호출		
	호출	호출	반환값
5	`Solve(5)`		
4		`Query({1,1,1,0,0})`	
2			2
5		`Answer({4,2,5,3,1})`	
3			(없음)
1			

이 문제에서 책이 놓인 방향이 왼쪽에서 오른쪽인지, 오른쪽에서 왼쪽인지의 방향은 신경 쓰지 않는다. 그렇기 때문에, 배열을 거꾸로 쓴 `Answer({1,3,5,2,4})`를 호출하여도 정답이다.