

## 팰린드롬 판별하기 (palindrome)

KOI사에서는 알고리즘 대회를 홍보하기 위한 새로운 이벤트를 만들었다! 이벤트에 참가하기 위해서는 KOI사만이 알고 있는 비밀 수열  $S$ 가 팰린드롬인지 판별해야 한다.

수열을 뒤집었을 때의 결과가 원래의 수열과 같아지는 수열을 **팰린드롬** 이라고 한다. 즉, 길이가  $N$ 인 수열  $S$ 가 팰린드롬이라는 것은, 모든  $0 \leq i \leq N - 1$ 에 대해  $S[i] = S[N - 1 - i]$ 이라는 것과 같다. 예를 들어,  $[1, 2, 3, 2, 1]$ ,  $[1, 2, 2, 1]$ 은 팰린드롬이지만  $[1, 2, 3, 1]$ ,  $[1, 2, 2]$ 는 팰린드롬이 아니다.

당신은 처음에 비밀 수열  $S$ 의 길이  $N$ 을 알고 있다. 또한,  $S$ 는 1 이상 5000 이하의 정수로 이루어진 수열임도 알고 있다. 이벤트 참가자들을 돕기 위해, KOI사는 특별 제작한 두 가지 기계를 제공한다.

- `count_pair` 기계에는 서로 다른 세 개의 수  $x, y, z$ 를 입력해야 한다. 이때 기계는  $S[x], S[y], S[z]$  중 같은 쌍의 개수를 반환한다. 예를 들어,  $S[x] = S[y] = S[z]$  일 경우 기계는 3을 반환한다.
- `find_character` 기계에는 하나의 정수  $x$ 와 정수들의 목록  $Y$ 를 입력해야 한다. 이때 기계는  $S[x] = S[y]$ 인  $y$ 가 목록  $Y$ 에 있다면 1, 없으면 0을 반환한다.
- 두 가지 기계에 입력되는 모든 수는 반드시 0 이상  $N - 1$  이하의 정수여야 한다.
- `find_character` 기계에 입력한  $Y$ 의 크기의 합은  $N$  이하여야 한다.

당신은 적은 횟수로 기계를 사용하여 비밀 수열  $S$ 가 팰린드롬인지 판별해야 한다.

## 함수 목록 및 정의

다음 함수를 구현해야 한다:

```
int guess_palindromicity(int N)
```

- $N$ : 수열  $S$ 의 길이
- 이 함수는  $S$ 가 팰린드롬이라면 1, 아니라면 0을 반환해야 한다.
- 이 함수는 한 개의 테스트 케이스에서 1번 이상 호출되며, 여러 번 호출될 수도 있다.

프로그램은 다음 함수를 호출할 수 있다:

```
int count_pair(int x, int y, int z)
```

- $x, y, z$ 는 0 이상  $N - 1$  이하의 서로 다른 정수여야 한다.
- 이 함수는  $S[x], S[y], S[z]$ 에서 서로 같은 쌍의 개수를 반환한다.
- 각 `guess_palindromicity` 호출에서, 이 함수를  $2N$ 번을 초과하여 호출할 수 없다.

```
int find_character(int x, vector<int> Y)
```

- $x$  및  $Y$ 의 모든 원소는 0 이상  $N - 1$  이하의 정수여야 한다.
- 이 함수는  $S[x] = S[y]$ 인  $y \in Y$ 가 있을 경우 1, 아닐 경우 0을 반환한다.
- 각 `guess_palindromicity` 호출에서, 이 함수를  $N$ 번을 초과하여 호출할 수 없다.
- 각 `guess_palindromicity` 호출에서, 호출하는 이 함수의  $Y$ 의 크기의 합이  $N$  이하이어야 한다.

제출하는 소스 코드의 어느 부분에서도 입출력 함수를 실행해서는 안 된다.

## 제약 조건

- $5 \leq N \leq 5000$
- $1 \leq S[i] \leq 5000$  (모든  $0 \leq i \leq N - 1$ )
- 한 개의 테스트 케이스에서 주어지는  $N$ 의 합을  $M$ 이라 할 때,  $5 \leq M \leq 5000$

이 문제에서 그레이더는 적응적이지 않다(NOT adaptive). 이것은  $S$ 가 그레이더의 수행 초기에 고정되어 쿼리에 따라 변하지 않음을 의미한다.

## 부분문제

각각의 `guess_palindromicity` 호출마다 아래와 같은 방식으로 점수를 매긴다. 당신의 제출이 받는 점수는 모든 테스트 케이스의 `guess_palindromicity` 호출에서 얻은 점수의 최솟값이다.

각 `guess_palindromicity` 호출에서, `count_pair` 함수의 호출 횟수를  $A$ , `find_character` 함수의 호출 횟수를  $B$ 라고 하자.

프로그램이 정상적으로 종료하지 않거나, `guess_palindromicity`가 반환한 값이 잘못된 경우 0점이 부여된다. `guess_palindromicity`가 반환한 값이 맞았을 때 부여되는 점수는 아래의 표와 같다.

조건	점수
$A \leq 2N, 2 \leq B \leq N$	15
$N < A \leq 2N, B \leq 1$	50
$\lfloor \frac{N}{2} \rfloor + 2 < A \leq N, B \leq 1$	70
$A = \lfloor \frac{N}{2} \rfloor + 2, B \leq 1$	90
$A \leq \lfloor \frac{N}{2} \rfloor + 1, B \leq 1$	100

## 예제 1

$N = 6, S = [1, 2, 3, 1, 2, 1]$ 라고 하자. 그레이더가 `guess_palindromicity(6)`을 호출한다. 함수 호출의 예가 아래에 보여진다.

Call	Return
<code>count_pair(0, 1, 2)</code>	0
<code>count_pair(3, 4, 5)</code>	1
<code>count_pair(0, 3, 5)</code>	3
<code>find_character(2, {0, 1, 3, 4, 5})</code>	0
<code>find_character(1, {0, 2, 4})</code>	1

## Sample grader

Sample grader는 아래와 같은 형식으로 입력을 받는다.

- Line 1:  $N$
- Line 2:  $S[0] S[1] \dots S[N - 1]$

프로그램이 **Accepted**로 판정되면, 샘플 그레이더는 **Correct : A B** 를 출력한다. 여기서, A는 `count_pair` 를 사용한 횟수, B는 `find_character` 을 사용한 횟수이다.

프로그램이 **Wrong Answer**로 판정되면, 샘플 그레이더는 **Wrong : MSG** 를 출력한다. 여기서 MSG는 다음 중 하나이다.

- **Wrong Input** : 입력 형식이 잘못되었다.
- **Invalid Query** : 쿼리에 넣은 값이 잘못된 값이다.
- **Wrong Guess** :  $S$ 가 팰린드롬인데 `guess_palindromicity`가 0을 반환했거나, 그 반대인 경우이다.

Sample grader는 실제 채점에서 사용하는 그레이더와 다를 수 있음에 유의하라.