3개의 봉우리

Cordillera Oriental은 볼리비아를 가로지르는 안데스 산맥의 산지이다. 이 산지에는 N개의 봉우리가 한 줄로 위치해 있으며 0번부터 $N-1$번까지 번호가 매겨져 있다. 봉우리 i ($0 \le i < N$)의 **높이**는 정수 $H[i]$로 표시하며 $1 \le H[i] \le N-1$이다.

임의의 두 봉우리 i와 j ($0 \le i < j < N$)에 대해, 둘 사이의 **거리** $d(i, j)$는 $j - i$로 정의된다.

고대 잉카 전설에 따르면, 3개의 봉우리가 다음의 특별한 조건을 만족하면 **신화적**이라 한다. 그 특별한 조건은 3개의 봉우리의 높이가 **순서와 상관없이 (즉, 정렬해서)** 3개의 봉우리의 거리와 **일치**하는 것이다.

3개의 봉우리 (i, j, k)가 신화적일 조건을 엄밀하게 정의하면 다음과 같다.

- $0 \le i < j < k < N$,
- 봉우리들의 높이 $(H[i], H[j], H[k])$가 봉우리들의 거리 $(d(i, j), d(i, k), d(j, k))$와 순서와 상관없이 일치한다. 예를 들면, 봉우리 0, 1, 2에 대해서 봉우리들의 거리는 $(1, 2, 1)$이다. 봉우리들의 높이 $(H[0], H[1], H[2])$가 $(1, 1, 2)$, $(1, 2, 1)$, $(2, 1, 1)$인 경우에는 봉우리들의 거리와 일치한다. 그러나 봉우리들의 높이가 $(1, 2, 2)$ 이면 봉우리들의 거리 $(1, 2, 1)$과 일치하지 않는다.

이 문제는 **Part I**과 **Part II**로 나뉘어져 있으며 각 서브태스크는 Part I 이나 Part II에 속한다. 응시자들은 서브태스크를 임의의 순서로 해결해도 상관없다. 특히, Part I을 모두 해결한 다음 Part 2를 해결해야 한다는 제약은 **없다**.

Part I

응시자들은 산지의 정보가 주어졌을 때 신화적인 3개의 봉우리의 갯수를 세는 것이다.

Implementation Details

응시자는 다음 함수를 구현해야 한다.

```
long long count_triples(std::vector<int> H)
```

- H: 봉우리의 높이를 나타내는 길이가 N인 배열이다.
- 이 함수는 테스트 케이스마다 정확하게 1번만 수행된다.

이 함수는 주어진 산지에 들어있는 신화적인 3개의 봉우리들의 갯수 T를 반환해야 한다.

Constraints

- $3 \le N \le 200\,000$
- $1 \le H[i] \le N - 1$ $(0 \le i < N)$.

Subtasks

Part I에 포함된 모든 서브태스크 점수의 합은 70점이다.

서브태스크	점수	추가 조건
1	8	$N \le 100$
2	6	$H[i] \le 10$ $(0 \le i < N)$.
3	10	$N \le 2000$
4	11	높이는 감소하지 않음 즉, $H[i-1] \le H[i]$ $(1 \le i < N)$.
5	16	$N \le 50\,000$
6	19	추가 조건은 없음

Example

다음의 호출을 보자.

```
count_triples([4, 1, 4, 3, 2, 6, 1])
```

산지에는 신화적인 3개의 봉우리들이 다음과 같이 3개 존재한다:

- 3개의 봉우리 $(1, 3, 4)$에 대해, 높이 $(1, 3, 2)$은 거리 $(2, 3, 1)$과 일치한다.
- 3개의 봉우리 $(2, 3, 6)$에 대해, 높이 $(4, 3, 1)$은 거리 $(1, 4, 3)$과 일치한다.
- 3개의 봉우리 $(3, 4, 6)$에 대해, 높이 $(3, 2, 1)$은 거리 $(1, 3, 2)$와 일치한다.

따라서 함수는 3을 반환해야 한다.

3개의 봉우리 $(0, 2, 4)$는 신화적이지 않다. 그 이유는 높이 $(4, 4, 2)$가 거리 $(2, 4, 2)$와 일치하지 않기 때문이다.

Part II

응시자는 신화적인 3개의 봉우리를 많이 포함하는 산지를 만들어야 한다. Part II에는 6 개의 **output-only** 서브태스크가 포함되어 있으며 **부분 점수**를 받을 수 있다.

각 서브태스크에는 두개의 양의 정수 M과 K가 주어진다. 응시자는 **최대** M개의 봉우리를 가지는 산지를 만들어야 한다. 응시자가 만든 산지가 **최소** K개의 신화적인 3개의 봉우리를 포함하면 해당 서브태스크에 할당된 최

고점수를 획득한다. 그렇지 않으면 응시자가 만든 산지에 포함된 신화적인 3개의 봉우리의 갯수에 비례하는 점수를 획득한다.

응시자는 유효한 산지를 만들어야 한다. 예를 들면 응시자가 만든 산지에 N ($3 \le N \le M$)개의 봉우리가 있다면 봉우리들의 높이 $H[i]$ ($0 \le i < N$)는 $1 \le H[i] \le N - 1$ 을 만족하는 정수라야 한다.

Implementation Details

응시자들이 해답을 제출하는 방법은 아래의 경우에 따라 다르다.

- **Output 파일**
- **Procedure 호출**

응시자들이 **output 파일** 방식으로 제출할 때는 다음과 같은 형식으로 text 파일을 만들고 제출해야 한다.

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

응시자들이 **procedure 호출** 방식으로 제출할 때는 다음의 함수를 구현해야 한다.

```
std::vector<int> construct_range(int M, int K)
```

- M: 봉우리 갯수의 최댓값
- K: 원하는 신화적인 3개의 봉우리의 갯수
- 이 함수는 각 서브태스크마다 정확하게 1번만 호출된다.

이 함수는 봉우리의 높이를 저장하는 길이가 N인 배열 H를 반환해야 한다.

Subtasks and Scoring

Part II에 포함된 모든 서브태스크 점수의 합은 30점이다. 각 서브태스크에 대해서 고정된 M과 K값이 아래 테이블에 표시되어 있다.

서브태스크	점수	M	K
7	5	20	30
8	5	500	2000
9	5	5000	50 000
10	5	30 000	700 000
11	5	100 000	2 000 000
12	5	200 000	12 000 000

각 서브태스크에 대해서 응시자가 유효한 산지를 만들지 못하면 0점을 획득한다. (CMS에 `Output isn't correct`로 표시된다.).

그렇지 않으면, 응시자의 해답에 포함된 신화적인 3개의 봉우리의 갯수를 T라고 했을 때 이 서브태스크의 점수는

$$5 \cdot \min\left(1, \frac{T}{K}\right).$$

이다.

Sample Grader

Parts I 과 Part II 는 동일한 샘플 그레이더 프로그램을 사용하며 입력의 첫줄을 가지고 구분된다.

Part I 입력 포맷:

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

Part I 출력 포맷:

```
T
```

Part II 입력 포맷:

```
2
M K
```

Part II 출력 포맷:

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

샘플 그레이더의 출력은 Part II 의 출력 파일에 요구되는 포맷과 일치함에 유의하라.