# Problem 1671: Minimum Number of Removals to Make Mountain Array

## Problem Information

**Difficulty:** Hard
**Acceptance Rate:** 54.84%
**Paid Only:** No
**Tags:** Array, Binary Search, Dynamic Programming, Greedy

## Problem Description

You may recall that an array `arr` is a **mountain array** if and only if:

* `arr.length >= 3` * There exists some index `i` (**0-indexed**) with `0 < i < arr.length - 1` such that: * `arr[0] < arr[1] < ... < arr[i - 1] < arr[i]` * `arr[i] > arr[i + 1] > ... > arr[arr.length - 1]`

Given an integer array `nums`■■■, return _the**minimum** number of elements to remove to make _`nums _■■■_` ___a**mountain array**._

**Example 1:**

**Input:** nums = [1,3,1] **Output:** 0 **Explanation:** The array itself is a mountain array so we do not need to remove any elements.

**Example 2:**

**Input:** nums = [2,1,1,5,6,2,3,1] **Output:** 3 **Explanation:** One solution is to remove the elements at indices 0, 1, and 5, making the array nums = [1,5,6,3,1].

**Constraints:**

* `3 <= nums.length <= 1000` * `1 <= nums[i] <= 109` * It is guaranteed that you can make a mountain array out of `nums`.

## Code Snippets

### C++:

```
class Solution {
public:
int minimumMountainRemovals(vector<int>& nums) {

}
};
```

### Java:

```
class Solution {
public int minimumMountainRemovals(int[] nums) {

}
}
```

### Python3:

```
class Solution:
def minimumMountainRemovals(self, nums: List[int]) -> int:
```