Find Peak Index in a Mountain Array LeetCode

Given an array of integers representing a mountain, where the values increase and then decrease, find the index of the peak element.

Input: arr = [0,1,0]

Output: 1

Input: arr = [0,2,1,0]

Output: 1

Input: arr = [0,10,5,2]

Output: 1

Approach 1: **Find the peak index in a mountain array using Linear**

Start with initializing the answer as the first index.

Iterate through the array and update the answer if a larger element is found.

Return the peak index.

Time Complexity: O(n), where n is the size of the array. In the worst case, it needs to iterate through the entire array.

Space Complexity: O(1), as it uses a constant amount of additional space.

Approach 2: Find the peak index in a mountain array using Binary Search

Initialize the lower bound (low) as 0 and the upper bound (high) as the last index of the array.

Perform binary search iterations until the lower bound is less than or equal to the upper bound.

Calculate the middle index (mid) using low + (high - low) / 2.

Compare the element at mid with its neighboring elements.

If arr[mid] < arr[mid + 1], the peak is on the right side. Update low = mid + 1.

If arr[mid] < arr[mid - 1], the peak is on the left side. Update high = mid - 1.

If neither condition is met, arr[mid] is the peak element. Return mid.

Return mid as the peak index.

Time Complexity: O(log n), where n is the size of the array. Binary search reduces the search space by half in each iteration.

Space Complexity: O(1), as it uses a constant amount of additional space.

Note:

The linear search approach has a time complexity of O(n), while the binary search approach has a time complexity of O(log n).

The binary search approach is more efficient than the linear search approach for larger arrays.