162. Find Peak Element

Medium

7081130FavoriteShare

A peak element is an element that is greater than its neighbors.

Given an input array nums, where nums[i] ≠ nums[i+1], find a peak element and return its index.

The array may contain multiple peaks, in that case return the index to any one of the peaks is fine.

You may imagine that nums[-1] = nums[n] = -∞.

Example 1:

Input: nums = [1,2,3,1]  
Output: 2  
Explanation: 3 is a peak element and your function should return the index number 2.

Example 2:

Input: nums = [1,2,1,3,5,6,4]  
Output: 1 or 5   
Explanation: Your function can return either index number 1 where the peak element is 2,   
 or index number 5 where the peak element is 6.

class Solution {

public:

bool isPeak(vector<int> nums, int mid){

if(nums[mid-1]<nums[mid]&&nums[mid]>nums[mid+1]) return true;

return false;

}

bool isIncreasingR(vector<int>& nums, int x){

if(nums[x]<nums[x+1]) return true;

return false;

}

bool isIncreasingL(vector<int> nums, int x){

if(nums[x-1]>nums[x]) return true;

return false;

}

int findPeak(vector<int>& nums,int start, int end){

if(start>end) return -1;

int mid=(start+end)/2;

if(isPeak(nums,mid)) return mid;

if(isIncreasingR(nums,start)&&isIncreasingL(nums,mid)) return findPeak(nums,start,mid);

if(isIncreasingR(nums,mid)&&isIncreasingL(nums,end)) return findPeak(nums,mid,end);

return -1;

}

int findPeakElement(vector<int>& nums) {

if(nums.size()==0) return -1;

if(nums.size()==1) return 0;

if(nums[1]<nums[0]) return 0;

if(nums[nums.size()-1]>nums[nums.size()-2]) return nums.size()-1;

return findPeak(nums,0,nums.size()-1);

}

};

Success

[Details](https://leetcode.com/submissions/detail/211557188/)

Runtime: 8 ms, faster than 99.42% of C++ online submissions for Find Peak Element.

Memory Usage: 12.4 MB, less than 5.02% of C++ online submissions forFind Peak Element.