Problem 852: Peak Index in a Mountain Array

Problem Information

Difficulty: Medium
Acceptance Rate: 0.00%
Paid Only: No

Problem Description

You are given an integer

mountain

array

arr

of length

n

where the values increase to a

peak element

and then decrease.

Return the index of the peak element.

Your task is to solve it in

O(log(n))

time complexity.

Example 1:

Input:

arr = [0,1,0]

Output:

1

Example 2:

Input:

arr = [0,2,1,0]

Output:

1

Example 3:

Input:

arr = [0,10,5,2]

Output:

1

Constraints:

3 <= arr.length <= 10

5

0 <= arr[i] <= 10

6

arr

is

guaranteed

to be a mountain array.

Code Snippets

C++:

```
class Solution {
public:
int peakIndexInMountainArray(vector<int>& arr) {

}
};
```

Java:

```
class Solution {
public int peakIndexInMountainArray(int[] arr) {

}
}
```

Python3:

```
class Solution:
def peakIndexInMountainArray(self, arr: List[int]) -> int:
```

Python:

```
class Solution(object):
def peakIndexInMountainArray(self, arr):
"""
:type arr: List[int]
:rtype: int
"""
```

JavaScript:

```
/**
* @param {number[]} arr
* @return {number}
*/
var peakIndexInMountainArray = function(arr) {

};
```

TypeScript:

```
function peakIndexInMountainArray(arr: number[]): number {

};
```

C#:

```
public class Solution {
public int PeakIndexInMountainArray(int[] arr) {

}
}
```

C:

```
int peakIndexInMountainArray(int* arr, int arrSize) {

}
```

Go:

```
func peakIndexInMountainArray(arr []int) int {

}
```

Kotlin:

```
class Solution {
fun peakIndexInMountainArray(arr: IntArray): Int {

}
}
```

Swift:

```
class Solution {
func peakIndexInMountainArray(_ arr: [Int]) -> Int {

}
}
```

Rust:

```
impl Solution {
pub fn peak_index_in_mountain_array(arr: Vec<i32>) -> i32 {

}
}
```

Ruby:

```
# @param {Integer[]} arr
# @return {Integer}
def peak_index_in_mountain_array(arr)

end
```

PHP:

```
class Solution {

/**
* @param Integer[] $arr
* @return Integer
*/
function peakIndexInMountainArray($arr) {

}
}
```

Dart:

```
class Solution {
int peakIndexInMountainArray(List<int> arr) {

}
}
```

Scala:

```
object Solution {
def peakIndexInMountainArray(arr: Array[Int]): Int = {

}
}
```

Elixir:

```
defmodule Solution do
@spec peak_index_in_mountain_array(arr :: [integer]) :: integer
def peak_index_in_mountain_array(arr) do

end
end
```

Erlang:

```
-spec peak_index_in_mountain_array(Arr :: [integer()]) -> integer().
peak_index_in_mountain_array(Arr) ->
.
```

Racket:

```
(define/contract (peak-index-in-mountain-array arr)
(-> (listof exact-integer?) exact-integer?)
)
```

Solutions

C++ Solution:

```
/*
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
*
```

```
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public:
int peakIndexInMountainArray(vector<int>& arr) {

}
};
```

Java Solution:

```
/**
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

class Solution {
public int peakIndexInMountainArray(int[] arr) {

}
}
```

Python3 Solution:

```
"""
Problem: Peak Index in a Mountain Array
Difficulty: Medium
Tags: array, search

Approach: Use two pointers or sliding window technique
Time Complexity: O(n) or O(n log n)
Space Complexity: O(1) to O(n) depending on approach
"""
```

```
class Solution:
def peakIndexInMountainArray(self, arr: List[int]) -> int:
# TODO: Implement optimized solution
pass
```

Python Solution:

```
class Solution(object):
def peakIndexInMountainArray(self, arr):
"""
:type arr: List[int]
:rtype: int
"""
```

JavaScript Solution:

```
/**
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

/**
* @param {number[]} arr
* @return {number}
*/
var peakIndexInMountainArray = function(arr) {

};
```

TypeScript Solution:

```
/**
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
```

```
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

function peakIndexInMountainArray(arr: number[]): number {

};
```

C# Solution:

```
/*
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

public class Solution {
public int PeakIndexInMountainArray(int[] arr) {

}
}
```

C Solution:

```
/*
* Problem: Peak Index in a Mountain Array
* Difficulty: Medium
* Tags: array, search
*
* Approach: Use two pointers or sliding window technique
* Time Complexity: O(n) or O(n log n)
* Space Complexity: O(1) to O(n) depending on approach
*/

int peakIndexInMountainArray(int* arr, int arrSize) {
}
```

Go Solution:

```
// Problem: Peak Index in a Mountain Array
// Difficulty: Medium
// Tags: array, search
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

func peakIndexInMountainArray(arr []int) int {

}
```

Kotlin Solution:

```
class Solution {
fun peakIndexInMountainArray(arr: IntArray): Int {

}
}
```

Swift Solution:

```
class Solution {
func peakIndexInMountainArray(_ arr: [Int]) -> Int {

}
}
```

Rust Solution:

```
// Problem: Peak Index in a Mountain Array
// Difficulty: Medium
// Tags: array, search
//
// Approach: Use two pointers or sliding window technique
// Time Complexity: O(n) or O(n log n)
// Space Complexity: O(1) to O(n) depending on approach

impl Solution {
pub fn peak_index_in_mountain_array(arr: Vec<i32>) -> i32 {

}
}
```

Ruby Solution:

```
# @param {Integer[]} arr
# @return {Integer}
def peak_index_in_mountain_array(arr)

end
```

PHP Solution:

```
class Solution {

/**
* @param Integer[] $arr
* @return Integer
*/
function peakIndexInMountainArray($arr) {

}
}
```

Dart Solution:

```
class Solution {
int peakIndexInMountainArray(List<int> arr) {

}
}
```

Scala Solution:

```
object Solution {
def peakIndexInMountainArray(arr: Array[Int]): Int = {
}
}
```

Elixir Solution:

```
defmodule Solution do
@spec peak_index_in_mountain_array(arr :: [integer]) :: integer
def peak_index_in_mountain_array(arr) do

end
end
```

Erlang Solution:

```
-spec peak_index_in_mountain_array(Arr :: [integer()]) -> integer().
peak_index_in_mountain_array(Arr) ->
.
```

Racket Solution:

```
(define/contract (peak-index-in-mountain-array arr)
(-> (listof exact-integer?) exact-integer?)
)
```