

Our Solution(s)Run Code

Solution 1

```
1 // Copyright © 2020 AlgoExpert, LLC. All rights reserved.
2
3 package main
4
5 // O(n) time | O(1) space - where n is the length of the input :
6 func LongestPeak(array []int) int {
7     longestPeakLength := 0
8     i := 1
9     for i < len(array)-1 {
10         isPeak := array[i-1] < array[i] && array[i] > array[i+1]
11         if !isPeak {
12             i += 1
13             continue
14         }
15
16         leftIdx := i - 2
17         for leftIdx >= 0 && array[leftIdx] < array[leftIdx+1] {
18             leftIdx -= 1
19         }
20
21         rightIdx := i + 2
22         for rightIdx < len(array) && array[rightIdx] < array[rightId
23             rightIdx += 1
24         }
25         currentPeakLength := rightIdx - leftIdx - 1
26         if currentPeakLength > longestPeakLength {
27             longestPeakLength = currentPeakLength
28         }
29         i = rightIdx
30     }
31     return longestPeakLength
32 }
33
```

Our Tests

Your SolutionsRun Code

Solution 1Solution 2Solution 3

```
1 package main
2
3 func LongestPeak(array []int) int {
4     // Write your code here.
5     return -1
6 }
7
```

Custom OutputSubmit Code

```
18 return 1
19 print(get_max_value_recursively(array))
20
21
22 def get_max_value_recursively(array):
23     array = array[1:]
24     array = longestPath(array)
25     return array[0], array[1]
26
27
28 def get_max_value_recursively(array):
29     array = array[1:]
30     array = longestPath(array)
31     return array[0], array[1]
32
33
34 def get_max_value_recursively(array):
35     array = array[1:]
36     array = longestPath(array)
37     return array[0], array[1]
```

Run or submit code when you're ready.