AP Computer Science A@Beijing National Day School

Lab 11: `Mountain`

Due date: Friday, March 8, 2019
Instructor: Mr. Alwin Tareen
Total Points: 15

Task Overview

- Implement a program that determines if an array satisfies mountain properties.

Background

- An array of positive integer values has the **mountain** property if the elements are ordered such that successive values increase until a maximum value(the peak of the mountain) is reached, and then the successive values decrease. The `Mountain` class declaration shown below contains methods that can be used to determine if an array has the mountain property. You will implement two methods in the `Mountain` class.

```
1  public class Mountain
2  {
3      /** @param array an array of positive integer values.
4       * @param stop the last index to check.
5       * Precondition: 0 <= stop < array.length
6       * @return true if, for each j such that 0 <= j < stop, array[j] < array[j+1];
7       * false otherwise.
8       */
9      public static boolean isIncreasing(int[] array, int stop)
10     { /* implementation not shown */}
11
12     /** @param array an array of positive integer values.
13      * @param start the first index to check.
14      * Precondition: 0 <= start < array.length-1
15      * @return true if, for each j such that start <= j < array.length -1,
16      * array[j] > array[j+1];
17      * false otherwise.
18      */
19     public static boolean isDecreasing(int[] array, int start)
20     { /* implementation not shown */}
21
22     /** @param array an array of positive integer values.
23      * Precondition: array.length > 0
24      * @return the index of the first peak(local maximum) in the array, if it exists;
25      * -1 otherwise.
26      */
27     public static int getPeakIndex(int[] array)
28     { /* to be implemented in part (a) */}
29
30     /** @param array an array of positive integer values.
31      * Precondition: array.length > 0
32      * @return true if array contains values ordered as a mountain;
33      * false otherwise.
```

```
34      */
35     public static boolean isMountain(int[] array)
36     { /* to be implemented in part (b) */ }
37
38     // There may be instance variables, constructors, and methods that are not shown.
```

(a) Write the `Mountain` method `getPeakIndex`. Method `getPeakIndex` returns the index of the first peak found in the parameter `array`, if one exists. A peak is defined as an element whose value is greater than the value of the element immediately before it and is also greater than the value of the element immediately after it. Method `getPeakIndex` starts at the beginning of the array and returns the index of the first peak that is found, or it returns `-1` if no peak is found.

For example, the following table illustrates the results of several calls to `getPeakIndex`.

arr	getPeakIndex(arr)
{11, 22, 33, 22, 11}	2
{11, 22, 11, 22, 11}	1
{11, 22, 33, 55, 77}	-1
{99, 33, 55, 77, 120}	-1
{99, 33, 55, 77, 55}	3
{33, 22, 11}	-1

(b) Write the `Mountain` method `isMountain`. Method `isMountain` returns `true` if the values in the parameter `array` are ordered as a mountain; otherwise, it returns `false`. The values in `array` are ordered as a mountain, if all three of the following conditions hold.

- There must be a peak.
- The array elements with an index smaller than the peak's index must appear in increasing order.
- The array elements with an index larger than the peak's index must apper in decreasing order.

For example, the following table illustrates the results of several calls to `isMountain`.

arr	isMountain(arr)
{1, 2, 3, 2, 1}	true
{1, 2, 1, 2, 1}	false
{1, 2, 3, 1, 5}	false
{1, 4, 2, 1, 0}	true
{9, 3, 5, 7, 5}	false
{3, 2, 1}	false

In writing `isMountain`, assume that `getPeakIndex` works as specified, regardless of what you wrote in part (a).

Specification

The Information Box Which Includes Your Name[5 points]

- Type your English and Pinyin name into the `Author` field, where it says: `YOUR NAME HERE`

Determine Mountain Properties [10 points]

- Write a `Java` program in the file `Mountain.java` that determines if an array has a peak, and if an array is a mountain.
- You will write your solution in a class called: `public class Mountain` right below the place where it says: `YOUR CODE HERE`.
- Make sure that you run your `Java` program, and ensure that it is free of errors.

Testing

- The file `MountainJUnitTest.java` contains the `JUnit` test cases which verify the correct functionality of the program.

Submission

- Submit your `Java` program by uploading it to the `Web-CAT` automated grading platform: `http://ec2-54-65-207-33.ap-northeast-1.compute.amazonaws.com:8080/Web-CAT/WebObjects/Web-CAT.woa`