Recursion 1 Lab

100 points

# Recursion 1 Lab Description

Implement the following functions using a recursive approach. Do not use class or instance variables to control the recursion! Everything should be done via the parameters and return values. Do not use any type of repetition control in your code (e.g. for or while loops).

Think about the problems as described below and try to understand the base case(s) and the words that describe the recursion case. The base cases are listed before the recursion case in all these examples because its easier to describe and convert to code in this way. However, you usually need to think of the recursion case first – at least in some high level form.

## Starter Code

| public class Recursion {  public static boolean isPalindrome(String input) {  }  public static int countLetter( String input, char letter ) {  }  public static int maxValue( int[] list, int n ) {  }  } |
| --- |

## 

## Palindrome Description

### Method Signature

| public static boolean isPalindrome(String input) |
| --- |

Return true if input is a palindrome. A palindrome is a sequence of letters that is mirrored at its center. Examples: abba, acttca, aca, a. Each input line should contain a string to be tested; spaces are considered part of the string.

* **Base case 1:** The length of the string is 0 or 1; both qualify as a palindrome; return true.
* **Base case 2:** The first and last characters of the string do not match; this string is not a palindrome; return false.
* **Recursion case:** The first and last characters of the string are equal; return the result of calling palindrome with the string after removing the first and last characters.

## CountLetter Description

### Method Signature

| public static int countLetter( String input, char letter ) |
| --- |

Return the number of occurrences of the character "letter" in the string “input”.

* **Base case 1:** The length of the string is 0; there are no characters left in the string that match the letter parameter; return 0.
* **Recursion case**: If the first character doesn’t match the parameter letter, return the result of calling countLetter on the rest of the string; if it does match, return 1 + the result of calling countLetter on the rest of the string.

## 

## MaxValue Description

### Method Signature

| public static int maxValue( int[] list, int n ) |
| --- |

Find the max value in the array list[0,...,n-1]; n is the number of elements in the portion of the array being tested.

Hint: Math.max() is useful here

* **Base case**: n = 1; the only element of the array must be the max, so return it.
* **Recursion case**: For n > 1, return the larger of the “last” element of (this portion of) the array and the maxValue of everything before it in the array.