**Lab Assignment #3 – Using Recursion**

Due Date: By the dropbox deadline.

Purpose: The purpose of this Lab assignment is to:

1. Design recursive algorithms
2. Implement recursive methods in Java

References: Read the course’s text chapter 5 and the lecture slides. This material provides the necessary information that you need to complete the exercises.

**Instructions:**

You **MUST** create a short demo video of your solution. Do not show yourself in the video. Upload your video in your personal youtube account or google-drive account and share it with the instructor **only**. Do not share it publicly. During submission at the dropbox, **write the link of your video** in the **Comments** **box** (present near the bottom of the submission page). Next, create a zip file of your solution as mentioned below in section **Submission Rules**, upload that zip file, and submit.

You **must** name a relevant Eclipse project according to the following rule:

YourFirstname\_YourLastname\_COMP254\_Labnumber\_ExerciseNumber.

Example: If student name is John Smith, the name of Eclipse project for Ex1 of Lab1 should be **John\_Smith\_COMP254\_Lab1\_Ex1**

**Submission Rules:**

Compress all your Eclipse projects as a **single** **zip** filethat is named according to the following rule: YourFirstname\_YourLastname\_COMP254\_Labnumber.zip

Example: **John\_Smith\_COMP254\_Lab1.zip**

Submit the above single zip file using the procedure mentioned in section **Instructions** above.

**Evaluation:**

|  |  |
| --- | --- |
| **Correct implementation of requirements:**   * Correct recursive algorithm * Correct Java implementation * Explanation of algorithm | 90% |
| **Friendly I/O** | 10% |
| **Total** | 100% |

**Exercise 1**

Create a **recursive** Java method **getProduct** to compute the **product of two positive integers**, *m* and *n*, using only addition and subtraction. **Implement** the Java code. Test the method by calling it in the main method of the relevant class. **Hint:** You need subtraction to count down from *m* or *n* and addition to do the arithmetic needed to get the right answer. The **linearSum** method from **Lesson5Examples** posted in the **eCentennial** module “**Lesson Examples (from textbook)**” may be helpful.

(5 marks)

**Exercise 2**

Write a **recursive** Java method **isPalindrome** that determines if a string s is a palindrome, that is, it is equal to its reverse. Examples of palindromes include 'racecar' and 'aaabbaaa'. Test the method by asking the user to provide string entries to be checked. **Hint:** Check the equality of the first and last characters and recur (but be careful to return the correct value for both odd and even-length strings).

(5 marks)