CS1020 Sit-in Lab 04 10:00AM Session - Strings and recursion

Semester 2 AY2012/2013

Background

In this last sit-in lab, there will be 2 tasks, one easy and one hard(er). Please attempt the first task (the easy one) before moving on to the second task. Both task are well known problems associated with strings.

Note: You are required to make use of RECURSION to solve both task 1 and task 2.

Problem Description

Task 1 - String Reversal

Write a recursive function to output the reverse of a given string containing only uppercase letters. Example

```
APPLE -> ELPPA
```

Input

The string of uppercase letters.

Output

The reversed string.

Skeleton Program

For this task, your program is to be named **StringReversal.java** (do not change this name). A skeleton program is provided, but you can change it any way or add new class(es) as you deem fit when you write your program.

```
import java.util.*;

class StringReversal {
  public StringReversal(Scanner sc) {
  }

  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    StringReversal sr = new StringReversal(sc);
  }
}
```

Testing Your Program

Three test cases, stringreversal1.in, stringreversal2.in and stringreversal9.in are provided for you to test your program. To test StringReversal.java with say stringreversal1.in, you type:

```
java StringReversal < stringreversal1.in
```

Task 2 - Longest Common Subsequence

Given 2 strings of uppercase letters and which can be of different lengths, write a recursive function to compute the length of the longest common subsequence between them. For example given the following 2 strings,

```
AIKHIEF
FHAIVHE
```

the longest common subsequence is of length 4 as shown in the underlined characters. In general, there may be multiple longest common subsequences.

```
AIKHIEF
FHAIVHE
```

Input

The 2 strings, which comprise only uppercase letters.

Output

The LENGTH of the longest common subsequence between the 2 strings.

Skeleton Program

For this task, your program is to be named **LCS.java** (do not change this name). A skeleton program is provided, but you can change it any way or add new class(es) as you deem fit when you write your program.

```
import java.util.*;

class LCS {
  public LCS(Scanner sc) {
  }

  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    LCS lcs = new LCS(sc);

  }
}
```

Testing Your Program

Three test cases, lcs1.in, lcs2.in and lcs8.in are provided for you to test your program. To test LCS.java with say lcs1.in, you type:

```
java LCS < lcs1.in
```

Grading Scheme

1. Program Correctness = 80 marks,
   (a) 30 marks for task 1
   (b) 50 marks for task 2
2. Programming Style = 20 marks (10 marks for each program)
3. Substantial marks will be lost if program does not compile.
4. There are 10 test cases for each task.
   (a) Task 1 test case is worth 3 marks each
   (b) Task 2 test case is worth 5 marks each
5. No marks will be awarded if both programs do not implement a recursive solution.
6. Things to look out for under programming style
   (a) Meaningful comments (including pre and post condition description)
   (b) Modularity of program
   (c) Proper indentation
   (d) Meaningful identifiers