King Saud University
College of Computer and Information Sciences
Department of Computer Science
CSC113 – Computer Programming II – Recursion Lab – Fall 2019

Exercise: Solve the following problems using recursion and test your implementations on the provided examples.
Note: in some of these problems, you might need more than one method to solve one problem (one public for the user, the other private with more arguments hidden from the user).
Example:

```
public static void method(Object arg) {
      …
      methodRec(arg, 0);
}
private static void methodRec(Object arg, …){
      …
}
```

1. Write a static recursive method ***printString*** that takes in an array of characters and prints them as a string. Assume the array is not empty and ***position*** is always valid.
   Method signature: *public static void printString(char[] string, int position)*
   Examples:
   - string = {'L', 'a', 'b', '_', '9'}, after calling *printString(string, 0)* → (Lab_9.) is printed out
   - string = {'C', 'S', 'C', ' ', '1', '1', '3'}, after calling *printString(string, 4)* → (113.) is printed out
   - string = {'A'}, after calling *printString(string, 0)* → (A.) is printed out

2. Write a static recursive method ***isMutanader*** that takes in a string and returns true if the string is ***متناظر*** or false otherwise. A string is considered ***متناظر*** it can be read forwards and backwards the same way. Assume the string has no space.
   Method signature: *public static boolean isMutanader(String word)*
   Examples:
   - "kitab" → false
   - "" → true
   - "algebra" → false
   - "wow" → true
   - "baab" → true
   - null → false

3. Write a static recursive method ***copyStrings*** that takes in an array of strings and two positions ***from & to*** that specify the range of the indices to be copied from. The method should return an array of strings containing the copied strings. Assume ***from & to*** are within range.
   Method signature: *public static String[] copyStrings(String[] src, int from, int to)*
   Examples:
   - src = {"First", "Second", "Third", "Fourth"}, after calling *copyStrings(src, 1, 3)* → returned array = {"Second", "Third", "Fourth"}
   - src = {"First", "Second", "Third", "Fourth"}, after calling *copyStrings(src, 0, 0)* → returned array = {"First"}
   - src = {"First", "Second", "Third", "Fourth"}, after calling *copyStrings(src, 3, 2)* → returned array = null
   - src = {"First", "Second", "Third", "Fourth"}, after calling *copyStrings(src, 0, 3)* → returned array = {" First", "Second", "Third", "Fourth"}
   - src = {}, after calling *copyStrings(src, x, y)* → returned array = null
   - src = null, after calling *copyStrings(src, x, y)* → returned array = null

4. Write a static recursive method ***calculateRange*** that takes in an array of integers and returns the range of its values. The range is defined as the difference between the highest and lowest values in a set. The array is not necessarily sorted.
   Method signature: *public static Integer calculateRange(int[] set)*
   Examples:
   - set1 = {1, 2, 3, 4, 5, 6} → 5
   - set2 = {-3, 0, 2, 5, 10} → 13
   - set3 = {10, 12, 15, 20, 20, 20} → 10
   - set4 = {} → null
   - set5 = null → null