CS 304 Homework Assignment 3

Due: 11:59pm, Thursday, October 6th

This assignment is scored out of 65. It consists of 6 questions. When you submit, you are required to create a folder with your name (Last name first, then First name), CS304, HW3, e.g., LastName_FirstName_CS304_HW3. Type your answers into a text file (**only `.txt`, `.doc`, and `.pdf` file formats are accepted**) and save it in this folder. Put all your Java programs (`*.java`) as well as output files in the same folder. Zip this folder, and submit it as one file to Desire2Learn. Do not hand in any printouts. Triple check your assignment before you submit. **If you submit multiple times, only your latest version will be graded and its timestamp will be used to determine whether a late penalty should be applied.**

Short Answers

P1. (8pts, 1pt each) Q1 on page 229-230

P2. (4pts) Q10 on page 232

P3. (5pts) Q39 on page 239

P4. (4pts, 2pts each) Q6 – a, b on page 287 (You don't need to answer part c.)

P5. (6pts, 2pts each) Q7 on page 287

Programming Questions

P6. (38pts + 12pts bonus)

a. Completing the `RecursiveMethods` class

In the **`RecursiveMethods`** class, you are required to implement the following methods using recursive solutions (**no looping statements**):

`largestRec(int[] arr, int pos)` – This method takes an integer array as well as an integer (the starting index) and returns the largest number in the array.

`sumRec(int[] arr, int pos)` – This method takes an integer array as well as an integer (the starting index) and returns the sum of the values in the array.

`isSorted(int[] arr, int start, int end)` – This method takes an array as well as two integers (the starting index and the end index). It returns **`true`** if the array is sorted in nondecreasing order between the indices **`start`** and **`end`**, or **`false`** otherwise.

The following method is for bonus points so its implementation is not mandatory:

`reverseListRec(LNode head)` – This method takes a reference to the head of a linked list and returns the reference to the head of the linked list in the reversed order.

Note that you are only supposed to touch the above four methods. You are NOT allowed to create any other methods, instance variables, or make any changes to methods other than these four methods or files other than "`RecursiveMethods.java`". You should also NOT change the content of the array that is passed into each method. Points will be taken off if you fail to follow this rule.

b. Code Testing

You are provided with a test driver implemented by "**`TestRecursiveMethods.java`**" (**Do not make any changes to this file!**) so there is no need to write your own.

Once you have completed the methods, you can run the test. You should create a plain text file named "**`output.txt`**", copy and paste the output (if your code crashes or does not compile, copy and paste the error messages) to this file and save it.

Grading Rubrics:

Code does not compile: -10
Code compiles but crashes when executed: -5
Changes were made to things other than the required methods: -5
`largestRec` was implemented in a non-recursive way: -5
`largestRec` changes the content of the array parameter: -3
`sumRec` was implemented in a non-recursive way: -5
`sumRec` changes the contents of the array parameter: -3
`isSorted` was implemented in a non-recursive way: -5
`isSorted` changes the contents of the array parameter: -3
Has output file: 5
Code passes 11 test cases: 33 (each test case worth 3 points)

BONUS:
`reverseListRec` was implemented in a non-recursive way: -12
Code passes 4 bonus test cases: 12 (each test case worth 3 points)

Sample output:

```
Test 1: sum(10) ==> [Passed]
 Expected: 10
 Yours: 10

Test 2: sum(10, 20, 30, 40, 50, 60) ==> [Passed]
 Expected: 210
 Yours: 210

Test 3: largest(10) ==> [Passed]
 Expected: 10
 Yours: 10

...
```

```
Test 10: isSorted([3, 2, 1, 4, 5, 6, 7], 1, 4) - [Passed]
 Expected: false
 Yours: false

Test 11: isSorted([3, 2, 1, 4, 5, 6, 7], 2, 6) - [Passed]
 Expected: true
 Yours: true

Total test cases: 11
Correct: 11
Wrong: 0

================ Bonus Question ================
Test 1: reverseListRec(null) ==> [Passed]
 Expected: head->null
 Yours: head->null

Test 2: reverseListRec(20) ==> [Passed]
 Expected: head->20->null
 Yours: head->20->null

Test 3: reverseListRec(20->30->40) ==> [Passed]
 Expected: head->40->30->20->null
 Yours: head->40->30->20->null

Test 4: reverseListRec(90->80->70->60->50->40) ==> [Passed]
 Expected: head->40->50->60->70->80->90->null
 Yours: head->40->50->60->70->80->90->null

Total bonus test cases: 4
Correct: 4
Wrong: 0
```