CS 304 Homework Assignment 5

Due: 11:59pm, Thursday, November 10^{th}

This assignment is scored out of 58. 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, HW5, e.g., LastName_FirstName_CS304_HW5. 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. (7pts, 1pt each) Q9 on page 525 (Use as few method calls as possible.)

P2. (3pts) What is the purpose of using the header and trailer nodes in a linked list?

P3. (6pts, 2pts each) Q1 on page 598

P4. (7pts, 1pt each) Q8 on page 600 (List the actual letter sequences for e, f, and g.)

P5. (10pts) Draw a binary search tree whose element are inserted in the following order:

```
50 72 96 94 107 26 12 9 15 13
```

Programming Questions

P6. (25pts)

a. Completing the `BST` class

You are provided with the files "**`BSTNode.java`**" and "**`BST.java`**". You are required complete the methods in the latter file to implement a binary search tree. This tree is designed to store only integers and it does not allow duplicate elements. You need to write the following two methods in a **non-recursive way**:

`add(int v)` – This method takes an integer and inserts it into the binary search tree, keeping the tree ordered.

`inOrder()` – This method prints the content of the binary search tree using in-order traversal.

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 "`BST.java`". Points will be taken off if you fail to follow this rule.

b. Code Testing

You are provided with a test driver implemented by "**`TestBST.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
Has output file: 5
`add` was implemented in a recursive way: -5
`add` was correctly implemented: 10
`inOrder` was implemented in a recursive way: -5
`inOrder` was correctly implemented: 10

Sample Output:

```
Test 1:
The tree should have 10 nodes and your tree has 10 nodes.
The in-order traversal of the tree should print the following sequence: 3 5
10 15 17 20 25 30 33 40.
Your tree prints the following sequence: 3 5 10 15 17 20 25 30 33 40

Test 2:
The tree should have 9 nodes and your tree has 9 nodes.
The in-order traversal of the tree should print the following sequence: 10 20
30 40 50 60 70 80 90.
Your tree prints the following sequence: 10 20 30 40 50 60 70 80 90

Test 3:
The tree should have 5 nodes and your tree has 5 nodes.
The in-order traversal of the tree should print the following sequence: 10 20
30 40 50.
Your tree prints the following sequence: 10 20 30 40 50
```