COMP 282 Project 2: Height of Trees (30 points)

Due:

Idea: An ordinary binary search tree with n nodes will have height between $\lg n$ and n, while a red-black tree with n nodes will have a height between $\lg n$ and $2 \lg n$. The goal of the project is to determine the average heights of a ordinary binary search tree and a red-black tree when the values 1, 2, ..., n are inserted into the tree in all possible permutations.

Requirements: Write a java program (1 or more java classes) that asks for a value n (my tests will be with a value between 1 and 40 - I expect larger values to take a long time), calculates the heights of a BST and RBT for every possible permutation of inserts of 1 through n, and prints out the averages.

You should

- have a class called BST.java which implements the insert and height methods for BSTs (because of delays you do no have to implement find or delete.
- have a class called RBT.java which implements the insert and height methods for RBTs (again find and delete are not required).
- have a class called Driver2.java which asks for the value of n, runs a loop over all permutations of 1 through n inserting that permutation into an empty BST and RBT, calculating the heights of both (and accumulating them), and then prints out the average height.

You must also have a file called status.txt which contains your name and a short (2-10 sentence) description of the status of your program. This file should be an ascii file. Though you may create it with MS Word (or notepad/wordpad/jGrasp/etc), you should be certain that it is a text file. A sample status.txt file is below.

Details: Your program must compile with the command "javac *.java". In particular, this means that you should not place your classes into a package (or technically you should place them in the default package).

Your program must run with the command "java Driver2". In particular, this means that you should have a file Driver2.java which includes a main method.

Your program should match the format of the sample interaction below.

You should be using good programming style. At a minimum break the project into appropriate classes, place your name near the top of each file, comment appropriately, be limited to 80 character lines, be limited to 30 line methods (usually shorter), and be properly indented.

Submission: Prior to the deadline place all of your files into a directory with a descriptive name (eg I would use NogaProj2), zip that directory and upload it in canvas. I would suggest uploading long before the deadline and updating/replacing as you go (work on it today and upload, work on it tomorrow and replace, work on it the next day and replace, ...

Sample interactions: The following examples show the format of user prompts and output desired. As for values:
if n is 1 there is only 1 permutation and it yields a 1 noded BST/RBT with height 1
if n is 2 there are 2 permutations both yielding a BST/RBT with height 2
if n is 3 there are 6 permutations, 4 of them yield a BST with height 3, 2 of them yield a BST with height 2, and all yield a RBT with height 2.

```
Welcome to Project 2: Average Heights
Enter a value for n
> 1
The average height of a BST with 1 node is 1.
The average height of a RBT with 1 node is 1.
```

```
Welcome to Project2: Average Heights
Enter a value for n
```

```
> 2
The average height of a BST with 2 nodes is 2.
The average height of a RBT with 2 nodes is 2.
```

```
Welcome to Project2: Average Heights
Enter a value for n
> 3
The average height of a BST with 3 nodes is 2.6666
The average height of a RBT with 3 nodes is 2.
```

Sample status.txt file:

```
John Noga - Project 1
The program works as required. It compiles/runs and the output matches the
correct format to the letter. However, the style and formatting is incorrect
because I DIDN'T: comment it (didn't even put my name in the file), keep the
length of lines to 80 characters, and keep the length of methods to 30 lines.
```

Cheating: This project is an individual project. You can discuss this project with other students. You can explain what needs to be done and give suggestions on how to do it. You can use the web to find ideas. You cannot share source code with your fellow students or submit solutions written by someone else (including code downloaded from the internet).