IF184401 Design & Analysis of Algorithms (C)

Midterm Exam

Starting date:	30 March 2022
Deadline:	07 April 2022, 23:59 WIB. **Penalty: 0.15% of grade/minute of tardiness.**
Exam type:	Open, Individual Project
Send to:	
	MM Irfan Subakti <yifana@gmail.com>
	CC to Muhammad Afdal Abdallah <afdalabdallahm@ gmail.com>, Dicksen Alfersius Novian <dicksenan@gmail.com> & Adrian Santoso <mkadriansantoso@gmail.com>
	with the subject: IF184401_DAA(C)_MID_StudentID_Name
File type and format:	A full report of the *working file* (i.e., source code), output and analysis along with the *declaration* into 1 (one) .**ZIP** file.
Filename format:	IF184401_DAA(C)_MID_StudentID_Name.ZIP

Instruction
Please do these steps as in the following.

0. Download the *working template file* – and use it for your work, i.e., `IF184401_DAA(C)_MID_StudentID_Name.zip`. Use the following codes/files as the building blocks for the next steps. You have to implement these codes in `Java`. We use *Eclipse/NetBeans* as IDE for Java programming. The *working template file* has 1 folder: `MidtermExam`. Inside folder `MidtermExam`, the `src` folder will be found. Inside the `src` folder, there are two folders/`packages`:`mytree` and `mid`. Inside folder/`package`: `mid`, there are 3 files: `MyList.java`, `MyListOps.java` and `MyListOpsTest.java`. You can test class `MyList` and class `MyListOps` by running the file `MyListOpsTest.java`. Inside folder/`package`: `mytree`, there are 6 files: `MyTree.java`, `MyTreeOps.java`, `DAA1.java`, `DAA1Test.java`, `DAA2.java` and `DAA2Test.java`. Your task is to ***update*** and ***continue writing codes*** for the files **`DAA1.java`** and **`DAA2.java`**. You can test your code by running the files `DAA1Test.java` and `DAA2Test.java`.

1. **[20 points]** Based on `MyTree.java` and `MyTreeOps.java` above, please create a function, namely `isBST()` which has a *recursive* function inside this function. It checks whether a tree, i.e., `MyTree t`, is BST (*Binary Search Tree*). **Hint**: it is allowed to use a supported function for `isBST()`. Please update the function `isBST()` in file **`DAA1.java`**.

Function name: `public static boolean isBST (MyTree t)`
Supported function name:

```
private static Boolean isBST(MyTree t, int lowerBound, int
upperBound)
```

Supported function `isBST()` will be used to get a Boolean value whether `t` is BST or not where its value can be found between the range of `lowerBound` and `upperBound`.

2. **[10 points]** Please create a recursive function, namely `printDescending()` which receives an input of a BST `t` (where `t` is `MyTree` and its values are an integer), that be able to print the values of `t` in *descending order*. This function has to be created without making a separate list of values from `t`. Please update the function `printDescending()` in file **`DAA1.java`**.

Function name: `public static void printDescending(MyTree t)`

3. **[10 points]** Please create an efficient recursive function, namely `max()` which receives an input of a BST `t` (where `t` is `MyTree` and its values are an integer), that be able to get the maximum value of the `t`'s values. It is not allowed to traverse and to compare all of the nodes in the tree. However, you should traverse at most one path in the tree from the root. It means this function works in $O(\log n)$ time for BST. **Hint**: assume we are a node `x` in BST, then all of the values from the tree's left branches of `x` always have the less than or equal (≤) values compared to the value of node `x`. So, the maximum value won't be existing in the tree's left branches. Where is the maximum value? Please update the function `max()` in file **`DAA1.java`**.

Function name: `public static int max(MyTree t)`

4. **[10 points]** Please create a recursive function, namely `isHeightBalanced()` which receives an input `MyTree t`, that be able to check whether `t` has a balanced height (AVL tree condition). Please update the function `isHeightBalanced()` in file **`DAA2.java`**.

Function name: `public static Boolean isHeightBalanced(MyTree t)`

5. **[10 points]** AVL tree is a Height-Balanced (HB) tree. Please create a recursive function, namely `insertHB()` which receives the inputs of `int n` and `MyTree t`, that be able to insert `n` into `t` while it keeps preserving the AVL condition. Please update the function `insertHB()` in file **`DAA2.java.`**

Function name: `public static MyTree insertHB(int n, MyTree t)`

Hint: insertHB() has to used two supported functions, i.e., `rebalanceForLeft()` and `rebalanceForRight()`. These supported functions will also be used for the function `deleteHB()` in the following (see point 8).

`rebalanceForLeft()` is called when the left subtree of t may have grown taller by one notch. If it is indeed taller than the right subtree by two notches, return a height-balanced version of `t` using single or double rotations.
The subtrees of t are assumed to be already height-balanced, and no effort is made to rebalance them. Likewise, for the case of the right subtree, please use `rebalanceForRight()`.

6. **[15 points]** Function name: `private static MyTree rebalanceForLeft(MyTree t)`. Please update this function in file **`DAA2.java.`**

7. **[15 points]** Function name: `private static MyTree rebalanceForRight(MyTree t)`. Please update this function in file `DAA2.java`.

8. **[10 points]** Please create a recursive function, namely `deleteHB()` which receives the inputs of `MyTree t` and `int x`, that be able to delete `x` from `t` while it keeps preserving the AVL condition. Please update the function `deleteHB()` in file **`DAA2.java`**.

Function name: `public static MyTree deleteHB(MyTree t, int x)`

9. To avoid plagiarism/cheating, every student needs to pledge and declare, then she/he must submit her/his **signed pledge and declaration** as in the following. Failing to do so will be resulted in getting a 0 (zero) grade. Attach the **scanned**/**photo** of your *declaration* in your report.

   "By the name of Allah (God) Almighty, herewith I pledge and truly declare that I have solved midterm exam by myself, did not do any cheating by any means, did not do any plagiarism, and did not accept anybody's help by any means. I am going to accept all of the consequences by any means if it has proven that I have been done any cheating and/or plagiarism."

   [Place, e.g., Surabaya], [date, e.g., 07 April 2022]

   <Signed>

   [Full name, e.g., Srikanti Lumintu]
   [StudentID, e.g., 05112040000xxx]

10. Have a lovely day, guys! Good luck! 😊