

COMP 203 DATA STRUCTURES AND ALGORITHMS

HOMEWORK 3 (Total=100 points)

Deadline: 31.12.2023 23:59

Read the questions and rules carefully. They are clear and well defined.

Rules:

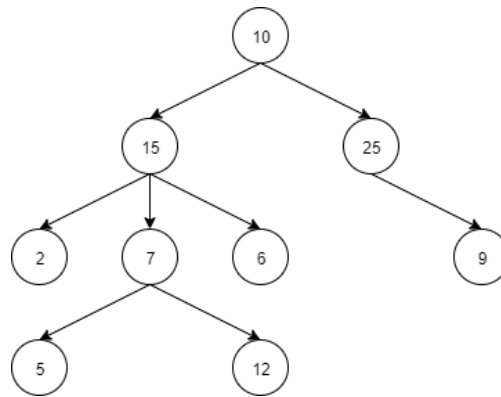
- 1. No Cheating:** You are not allowed to collaborate with your friends and use any kind of websites or AI. If your homework gives a sign of any of them, **directly it will be graded as zero.**
- 2. Goal:** Please do your homework alone. Our main aim is to **learn** whatever we cover so far.
- 3. Submission:** Submit your homework in **2 java files. No other file types will be accepted.** You will submit only 2 java files. **DON'T USE ZIP/RAR etc.** In these cases, your points will be deducted by 30%.
- 4. Coding policy:** Explain your code in comments. **This is a must!**
- 5. Latency policy:** A 10% deduction will be applied for each day of late submission.

QUESTIONS

1. Implement Tree abstract data structure using Node data structure. Implement the following classes and methods: **(55pt)**

Keep in mind that in a tree, a parent may have more than 2 children. That is why you should create children of a parent as a Queue data structure. Basically, a Node has "E data, Queue<Node<E>> childrenList, Node<E> parent".

1. Implement Node<E> class for Tree with constructor with the parameter E data. **(5pt)**
2. You can use Queue built in data structure of Java. You will use Queue data structure to save children of a parent. **(5pt)**
Hint: You can create childrenList as follows:
`Queue<Node<E>> childrenList =new LinkedList<>();`
3. Implement Integer QTree<E> class with the class variables and with no parameter constructor. **(5pt)**
4. deleteNode(Node<E> root, E deletedValue) that deletes the node with the given value *deletedValue* from the tree. **(10pt)**
5. Node<E> Find(Node<E> root, E value) that finds the node having the given *value* in the given tree. If it is found, it returns the found node else it returns null. **(10pt)**
6. Test your methods in the main by creating the following tree with the integer data type. (Creating the given tree: **5 pt**, testing 3 methods: **15 pt**)



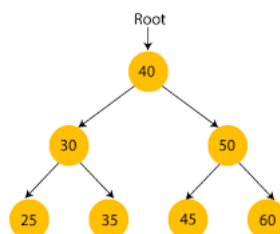
**Include comments of your code for each method and class.
Submit QTree.java to Canvas.**

Hint: You may use
 import java.util.LinkedList;
 import java.util.Queue;" to use Queue in your code.

2. Implement Binary Search Tree abstract data structure using Node data structure.
 Implement the following classes and methods: **(45pt)**

1. Implement Node<E> class for Binary tree with constructor with the parameter E data. **(5pt)**
2. Implement Integer BinarySearchTree <E> class with the class variables and with no parameter constructor. **(5pt)**
3. insert(Node<E> root, E value) that adds a new node with the given *value* to the given tree. **(10pt)**
4. delete(Node<E> root, E value) that deletes the node having the given *value*. **(10pt)**

Test your methods in the main by creating the following binary search tree with the integer data type. (Creating the given tree: **5 pt**, testing 2 methods: **10 pt**)



**Include comments of your code for each method and class.
Submit BinarySearchTree.java to Canvas.**