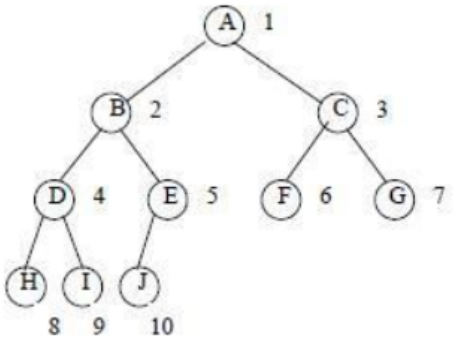
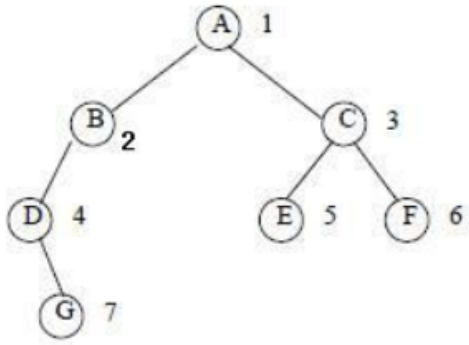
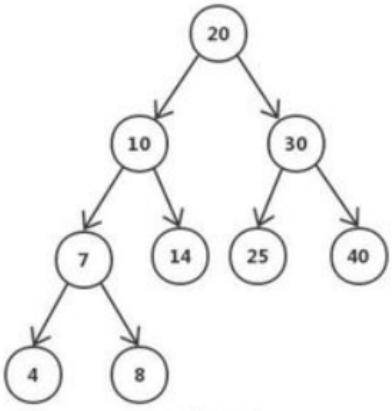
Data Structures and Algorithms
INFO 6205
Homework 5
Due: February 17, 2019

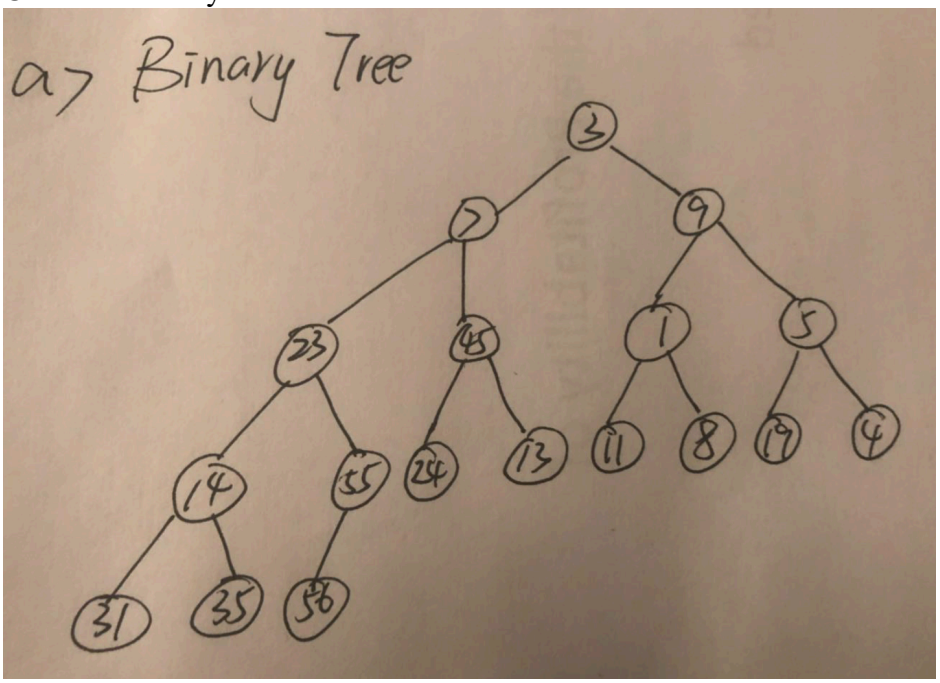
Put all your java, compiled class files and documentation files into a zip file named Homework5.zip and submit it via the drop box on the blackboard before the END of due date. Put your name on all .java files. There will be a short quiz on this homework.

1. What is the Balanced Tree, Complete Tree and Non-Complete Tree?

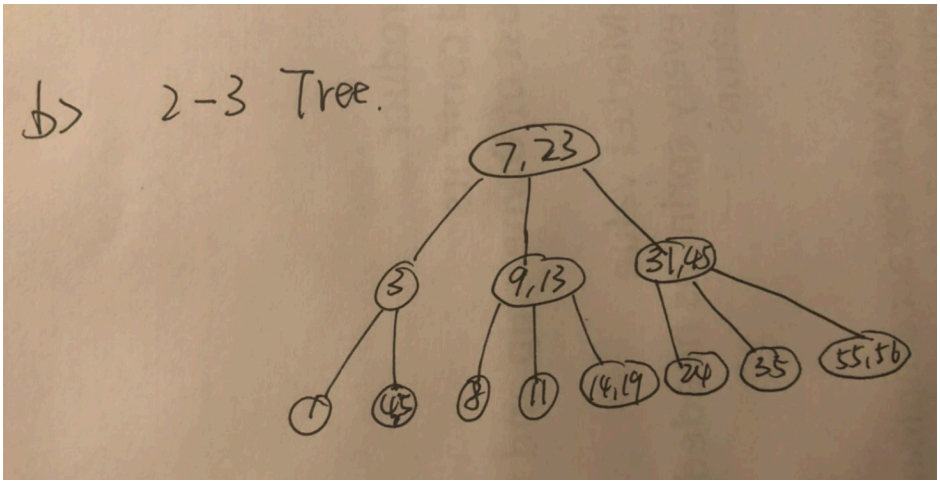
Tree Name	Definition	Example
Complete Tree	If the depth of the binary tree is h, except for the H th layer, the number of nodes in the other layers (1~H-1) reaches the maximum number, and all the nodes in the H th layer are continuously concentrated on the leftmost side, which is Complete Tree.	
Non-Complete Tree	The tree is not a full tree, and it doesn't belong to the Complete Tree.	
Balanced Tree	For any node in the Balanced Tree, the absolute value of the difference between the height of the left subtree of the node and the height of the right subtree is less than 2.	

2. Consider following data: {3,7,9,23,45,1,5,14,55,24,13,11,8,19,4,31,35,56}

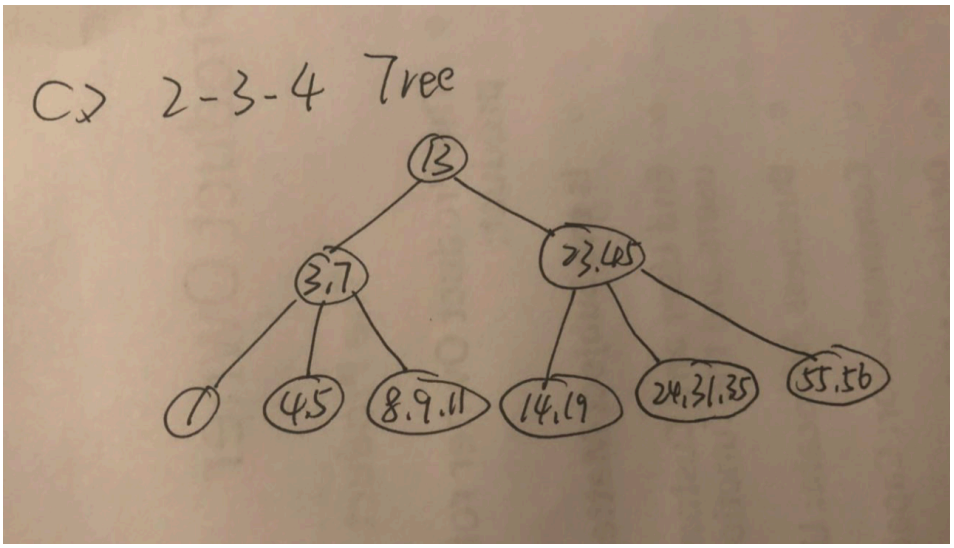
a) Construct Binary Tree



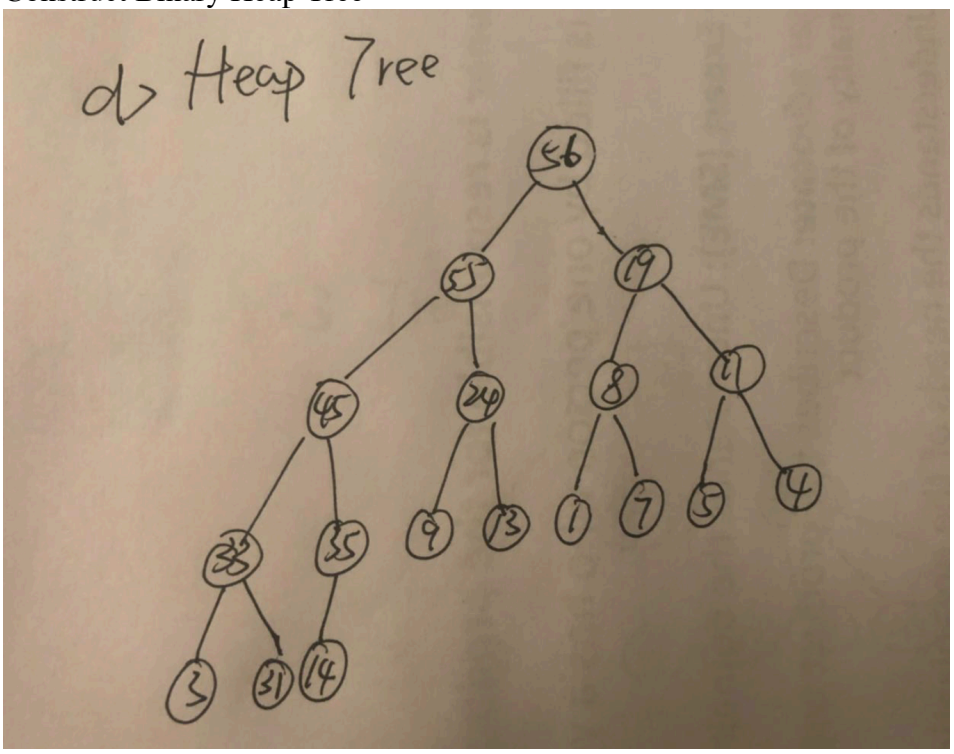
b) Construct 2-3 Tree



c) Construct 2-3-4 Tree



d) Construct Binary Heap Tree



e) What is Time complexity of each case, and Why would you use one versus the othe

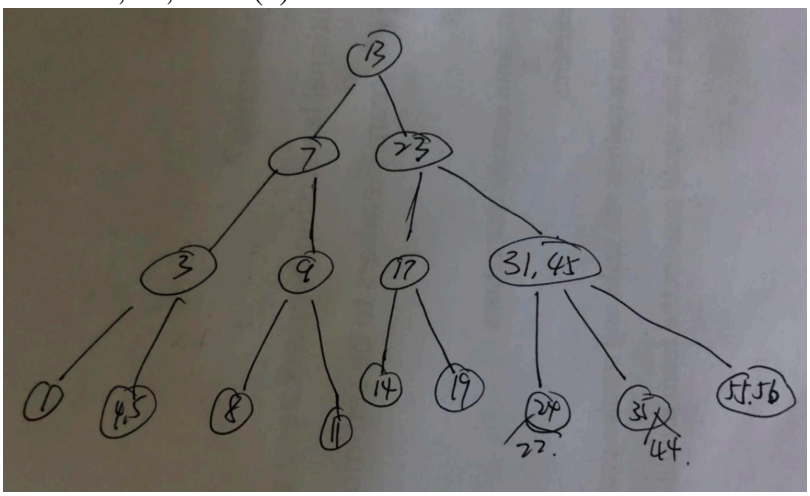
	A(Binary Tree)	B(2-3 Tree)	C(2-3-4 Tree)	D(Binary Heap Tree)
Time complexity	O(log2n)	O (log N)	O (log N)	*O(log2(n))*

f) Perform Inorder, Preorder, and Postorder traversal for (a), (b), (c)

Binary Tree	Preorder: 3, 7, 23, 14, 31, 35, 55, 56, 45, 24, 13, 9, 1, 11, 8, 5 , 19, 4

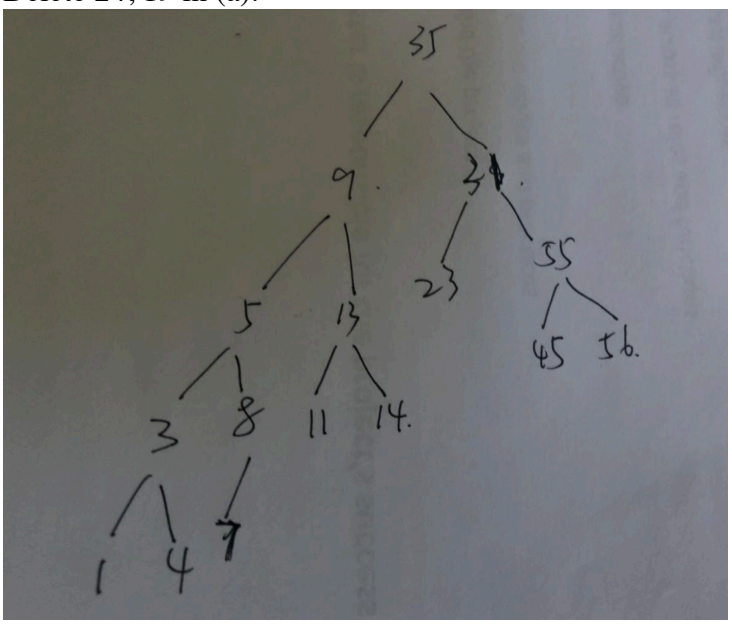
	Inorder: 31, 14, 35, 23, 56, 55, 7, 24, 45, 13, 3, 11, 1, 8, 9, 19, 5, 4 Postorder: 31, 35, 14, 56, 55, 23, 24, 13, 45, 7, 11, 8, 1, 19, 4, 5, 9, 3
2-3 Tree	Preorder: 7, 3, 1, 4, 5, 9, 8, 11, 13, 14, 19, 23, 31, 24, 35, 45, 55, 56 Inorder: 1, 3, 4, 5, 7, 8, 9, 11, 13, 14, 19, 23, 24, 31, 35, 45, 55, 56 Postorder: 1, 4, 5, 3, 8, 11, 9, 14, 19, 13, 7, 24, 35, 31, 55, 56, 45, 23
2-3-4 Tree	Preorder: 13, 3, 1, 7, 4, 5, 8, 9, 11, 23, 14, 19, 45, 24, 31, 35, 55, 56 Inorder: 1, 3, 4, 5, 7, 8, 9, 11, 13, 14, 19, 23, 24, 31, 35, 45, 55, 56 Postorder:1, 4, 5, 3, 8, 9, 11, 3, 7, 14, 19, 24, 31, 35, 23, 55, 56, 45, 13

f) Insert 17, 22, 44 in (b)

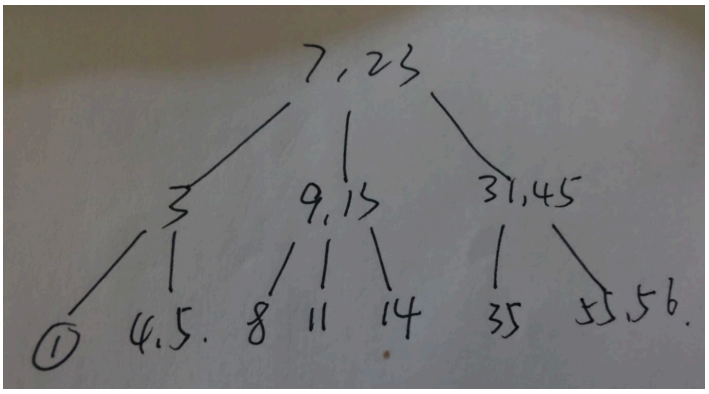


g) Delete 24, 19 in (a), (b), (c)

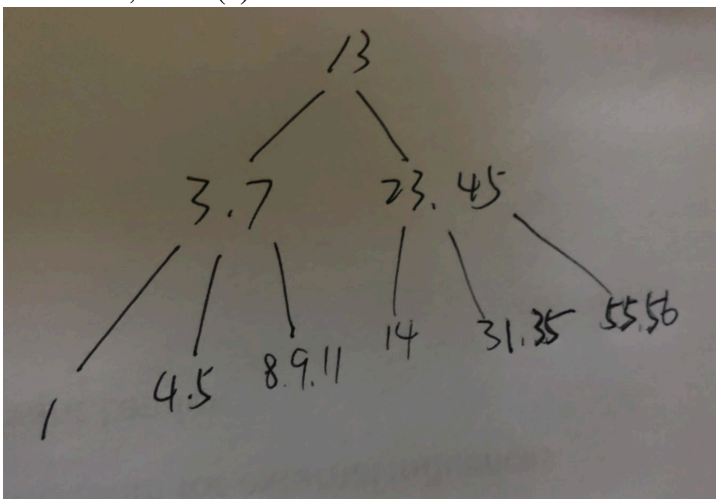
Delete 24, 19 in (a):



Delete 24, 19 in (b):



Delete 24, 19 in (c):



i) What is the Height of (a), (b), (c)?

	A(Binary Tree)	B(2-3 Tree)	C(2-3-4 Tree)	D(Binary Heap Tree)
Height	4	2	2	4

j) Write Java code for Search and Insert in (a), (b), (c)

k) Write Java code for DeleteMin() and Rank() Algorithms for (a), provide an example

3. Class Record is described below. Write Java code to build Binary Tree, 2-3 Tree and B-Tree where "key" is the value in data presented in problem-2.

```
public class Record {
    private int key
     private Node leftNode;
     private Node rightNode;

     public Record(int key, Node leftNode, Node rightNode) {
          this.key = key;
          this.leftNode = leftNode;
          this.rightNode = rightNode;
     }
```

```
        public Record(int key){
            this.key = key;
        }

        public int getKey() {
                return key;
        }

        public Node getLeftNode() {
                return leftNode;
        }

        public Node getRightNode() {
                return rightNode;
        }

        public void setKey(int key) {
                this.key = key;
        }

        public void setLeftNode(Node leftNode) {
                this.leftNode = leftNode;
        }

        public void setRightNode(Node rightNode) {
                this.rightNode = rightNode;
        }
}
```