

Mohammed Chowdhury
323.33
Q25

TREENODE CLASS:

```
1      public class treeNode {  
2          treeNode left;  
3          treeNode right;  
4          int data;  
5  
6          public treeNode(int data){  
7              this.data = data;  
8          }  
9      }  
10
```

BinTree Class:

```

8      public class binTree {
9          treeNode root;
10         int num;
11         int[] inAry;
12
13         int readCheck(String inFile){
14             int i = 0;
15             int prev = Integer.MIN_VALUE;
16             int count = 0;
17             try {
18
19                 File input = new File(inFile);
20                 Scanner scan = new Scanner(input);
21
22                 while(scan.hasNext()) {
23                     i = Integer.parseInt(scan.next());
24                     if (i > prev) {
25                         prev = i;
26                         count++;
27                     } else {
28                         return -1;
29                     }
30
31                 }
32             } catch (FileNotFoundException e){
33                 System.out.println("File not found. Make sure name is typed correctly");
34             }
35         }
36         return count;
37     }

```

```

38
39 void loadAry(String inFile, int[] inAry){
40     int i = 0;
41     try {
42         Stack<Integer> temp = new Stack<>();
43         File input = new File(inFile);
44         Scanner scan = new Scanner(input);
45         while(scan.hasNext()) {
46             i = Integer.parseInt(scan.next());
47             temp.push(i);
48         }
49         for(int j =0; j < inAry.length;j++){
50             inAry[j] = temp.pop();
51         }
52     }catch (FileNotFoundException e){
53         System.out.println("File not found. Make sure name is typed correctly");
54     }
55
56 }
57 @ void printAry(int[] inAry, FileWriter debug){
58     try {
59         for(int i =0; i < inAry.length; i++){
60             debug.write( str: inAry[i]+ " ");
61         }
62     }catch (IOException e){
63         System.out.println("error while printing inAry");
64     }
65 }

```

```

66     treeNode buildBinTree(int[] inAry, int leftIndex, int rightIndex){
67
68         //requirements say "until left index is less or equal to right index
69         //but leftIndex is always smaller from the start and it will hit
70         // base case, so i changed it to leftIndex > rightIndex
71         if(leftIndex > rightIndex){
72             return null;
73         }
74
75         int rootLocation = (leftIndex + rightIndex) / 2;
76         treeNode rootNode = new treeNode(inAry[rootLocation]);
77         rootNode.left = buildBinTree(inAry, leftIndex, rightIndex: rootLocation-1);
78         rootNode.right = buildBinTree(inAry, leftIndex: rootLocation+1, rightIndex);
79
80         return rootNode;
81     }
82
83     void preOrder(treeNode root, FileWriter treeFile){
84         if(root == null){
85             return;
86         }
87         try {
88             treeFile.write( str: root.data + "\n");
89         }catch (IOException e){
90             System.out.println("error in preorder");
91         }
92         preOrder(root.left, treeFile);
93         preOrder(root.right, treeFile);
94     }
95

```

```
96 void inOrder(treeNode root, FileWriter treeFile){
97     if(root == null){
98         return;
99     }
100
101     inOrder(root.left,treeFile);
102     try {
103         treeFile.write( str: root.data + "\n");
104     }catch (IOException e){
105         System.out.println("error in preorder");
106     }
107     inOrder(root.right,treeFile);
108 }
109
110 void postOrder(treeNode root, FileWriter treeFile){
111     if(root == null){
112         return;
113     }
114
115     postOrder(root.left,treeFile);
116     postOrder(root.right,treeFile);
117     try {
118         treeFile.write( str: root.data + "\n");
119     }catch (IOException e){
120         System.out.println("error in preorder");
121     }
122 }
123
124 }
```

treeFile:

PREORDER:

56
81
91
94
100
95
93
92
87
90
89
85
83
70
76
80
78
74
72
65
69
67
60
63
58
32
44
50
54
52
48
46
38
42
40
36
34
20
26
30
28
24

22
14
18
16
10
12
8

INORDER:

100
95
94
93
92
91
90
89
87
85
83
81
80
78
76
74
72
70
69
67
65
63
60
58
56
54
52
50
48
46
44
42
40
38
36
34

32
30
28
26
24
22
20
18
16
14
12
10
8

POSTORDER:

95
100
92
93
94
89
90
83
85
87
91
78
80
72
74
76
67
69
63
58
60
65
70
81
52
54
46
48
50
40

42
34
36
38
44
28
30
22
24
26
16
18
12
8
10
14
20
32
56

DebugFile:

Input Size: 49

ARRAY:

100	95	94	93	92	91	90	89	87	85	83	81	80	78	76	74	72
70	69	67	65	63	60	58	56	54	52	50	48	46	44	42	40	38
36	34	32	30	28	26	24	22	20	18	16	14	12	10	8		