

```
1  import java.io.*;
2  import java.util.*;
3
4  class BTNode{
5
6      private int nodeId;
7      private int data;
8      private int levelNum;
9      private BTNode leftChildPtr;
10     private BTNode rightChildPtr;
11
12     public BTNode() {}
13
14     public void setNodeId(int id){
15         nodeId = id;
16     }
17
18     public int getNodeId(){
19         return nodeId;
20     }
21
22     public void setData(int d){
23         data = d;
24     }
25
26     public int getData(){
27         return data;
28     }
29
30     public void setLevelNum(int level){
31         levelNum = level;
32     }
33
34     public int getLevelNum(){
35         return levelNum;
36     }
37
38     public void setLeftChildPtr(BTNode ptr){
39         leftChildPtr = ptr;
40     }
41
42     public void setRightChildPtr(BTNode ptr){
43         rightChildPtr = ptr;
44     }
45
46     public BTNode getLeftChildPtr(){
47         return leftChildPtr;
48     }
49
50     public BTNode getRightChildPtr(){
51         return rightChildPtr;
52     }
53
54     public int getLeftChildID(){
55         if (leftChildPtr == null)
56             return -1;
57
58         return leftChildPtr.getNodeId();
59     }
60
61     public int getRightChildID(){
62         if (rightChildPtr == null)
63             return -1;
64     }
```

```

65     return rightChildPtr.getNodeId();
66 }
67 }
68
69
70
71
72
73
74 class BinaryTree{
75
76     private int numNodes;
77     private BTNode arrayOfBTNodes[];
78
79     public BinaryTree(int n){
80         numNodes = n;
81         arrayOfBTNodes = new BTNode[numNodes];
82
83         for (int id = 0; id < numNodes; id++){
84             arrayOfBTNodes[id] = new BTNode();
85             arrayOfBTNodes[id].setNodeId(id);
86             arrayOfBTNodes[id].setLevelNum(-1);
87             arrayOfBTNodes[id].setLeftChildPtr(null);
88             arrayOfBTNodes[id].setRightChildPtr(null);
89         }
90     }
91
92     public void setLeftLink(int upstreamNodeID, int downstreamNodeID){
93         arrayOfBTNodes[upstreamNodeID].setLeftChildPtr(arrayOfBTNodes[downstreamNodeID]);
94     }
95
96     public void setRightLink(int upstreamNodeID, int downstreamNodeID){
97
98         arrayOfBTNodes[upstreamNodeID].setRightChildPtr(arrayOfBTNodes[downstreamNodeID]);
99         ;
100     }
101
102     public void printLeafNodes(){
103
104         for (int id = 0; id < numNodes; id++){
105
106             if (arrayOfBTNodes[id].getLeftChildPtr() == null &&
107                 arrayOfBTNodes[id].getRightChildPtr() == null)
108                 System.out.print(id + " ");
109         }
110
111         System.out.println();
112     }
113
114     public boolean isLeafNode(int nodeid){
115
116         if (arrayOfBTNodes[nodeid].getLeftChildPtr() == null &&
117             arrayOfBTNodes[nodeid].getRightChildPtr() == null)
118             return true;
119
120         return false;
121     }
122
123     public int getNodeHeight(int nodeid){
124
125         if (nodeid == -1)

```

```

125         return -1;
126
127     if (isLeafNode(nodeid) )
128         return 0;
129
130     int leftChildID = arrayOfBTNodes[nodeid].getLeftChildID(); // -1 if not exist
131     int rightChildID = arrayOfBTNodes[nodeid].getRightChildID(); // -1 if not exist
132
133     return Math.max(getNodeHeight(leftChildID), getNodeHeight(rightChildID)) + 1;
134
135 }
136
137
138 public int getTreeHeight(){
139     return getNodeHeight(0);
140 }
141
142
143 }
144
145
146 class BinaryTreeImplementation{
147
148     public static void main(String[] args){
149
150         try{
151
152             Scanner input = new Scanner(System.in);
153
154             String filename;
155             System.out.print("Enter a file name: ");
156             filename = input.next();
157
158             int numNodes;
159             System.out.print("Enter number of nodes: ");
160             numNodes = input.nextInt();
161
162             BinaryTree binaryTree = new BinaryTree(numNodes);
163
164             FileReader fr = new FileReader(filename);
165             BufferedReader br = new BufferedReader(fr);
166
167             String line = null;
168
169             while ( (line = br.readLine()) != null){
170
171                 StringTokenizer stk = new StringTokenizer(line, ",: ");
172
173                 int upstreamNodeID = Integer.parseInt(stk.nextToken());
174
175                 int childIndex = 0;
176
177                 while (stk.hasMoreTokens()){
178
179                     int downstreamNodeID = Integer.parseInt(stk.nextToken());
180
181                     if (childIndex == 0 && downstreamNodeID != -1)
182                         binaryTree.setLeftLink(upstreamNodeID, downstreamNodeID);
183
184                     if (childIndex == 1 && downstreamNodeID != -1)
185                         binaryTree.setRightLink(upstreamNodeID, downstreamNodeID);
186
187                     childIndex++;
188

```

```
189         }
190     }
191 }
192
193
194     System.out.print("Leaf Nodes: ");
195     binaryTree.printLeafNodes();
196     System.out.println();
197
198     System.out.println("Tree Height: " + binaryTree.getTreeHeight());
199     System.out.println("Height of node 1: " + binaryTree.getNodeHeight(1));
200
201 }
202 catch (Exception e) {e.printStackTrace();}
203
204 }
205 }
```

Enter a file name: binaryTreeFile_1.txt

Enter number of nodes: 10

Leaf Nodes: 5 6 8 9

Tree Height: 4

Height of node 1: 2