DSA Week 16 activities

This week, you are required to complete two questionnaires and two labs.

a. In this print out, answer all Week 16 questions.
b. Also, in this print out, complete Week 16 lab 1 & 2 using the lab computers.

Note: You can complete the activities in any order, however, make afford to complete and understand everything which prepares you for well for the Final Exam.

DSA Week 16 Questions

1. Differentiate between a general tree and binary tree data structure.

2. Discuss a real-world application of tree ADT in computing.

3. Discuss any two methods used with the tree ADT.

4. What is a child node, parent node, root node, internal node, external nodes, leave nodes and siblings node?

DSA Week 16 Lab Activity (Week11Lab1)

Using the lab computers create the following Java program using jGrasp!

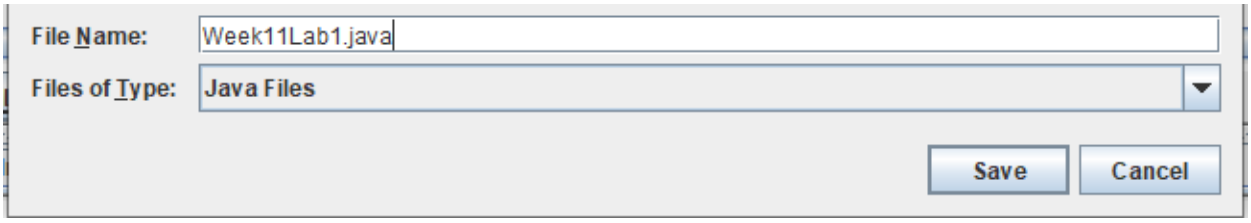
Step 1: Login to your lab computer and create a new java file in jGrasp.

Step 2: When the window below appears. Type the following code into jGrasp.

```
1  /* DSA Week 11 Lab 1 */
2
3  //import Arraylist and list classes
4  import java.util.ArrayList;
5  import java.util.List;
6
7  class TreeNode {
8      String value;
9      List<TreeNode> children;
10
11     TreeNode(String value) {
12         this.value = value;
13         this.children = new ArrayList<>();
14     }
15
16     void addChild(TreeNode child) {
17         children.add(child);
18     }
19 }
20
21 public class Week11Lab1 {
22
23     static void printTree(TreeNode node, int level) {
24         // Indent based on tree level
25         for (int i = 0; i < level; i++) {
26             System.out.print("    ");
27         }
28
29         System.out.println(node.value);
30
31         for (TreeNode child : node.children) {
32             printTree(child, level + 1);
33         }
34     }
35
```

```
36      public static void main(String[] args) {
37          // Create the root node
38          TreeNode root = new TreeNode("Diploma Programs");
39
40          // Add children in the same order as stack push
41          TreeNode node1 = new TreeNode("DIT");
42          root.addChild(node1);
43
44          TreeNode child1 = new TreeNode("D1");
45          node1.addChild(child1);
46
47          TreeNode child2 = new TreeNode("D2");
48          node1.addChild(child2);
49
50          TreeNode node2 = new TreeNode("DHRM");
51          root.addChild(node2);
52
53          TreeNode node3 = new TreeNode("DACC");
54          root.addChild(node3);
55
56          TreeNode node4 = new TreeNode("DICT");
57          root.addChild(node4);
58
59          // Print tree recursively
60          printTree(root, 0);
61      }
62  }
```

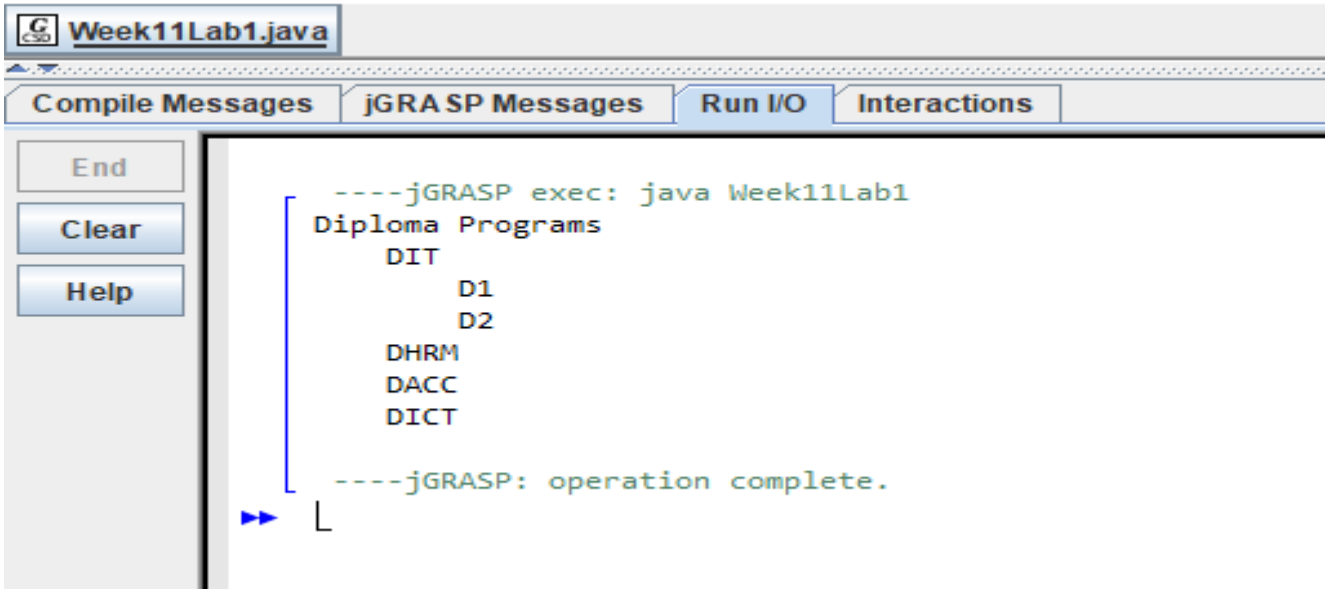
Step 3: Go to **file/save** to save your java program as **Week11Lab1**



Step 4: After saving, compile (**click on compile icon or on your keyword hold Ctrl + B**) to check for syntax errors.

Step 5: If compiling is successfully then run (**click on the find and run main method icon or on your keyboard hold Ctrl + R**) your program.

Step 6: If run is successful then you should see the following output in the console



Step 7: Week11Lab1 Completed! You have created your first tree data structure program.

DSA Week 16 Lab Activity (Week11Lab2)

Using the lab computers create the following Java program using jGrasp!

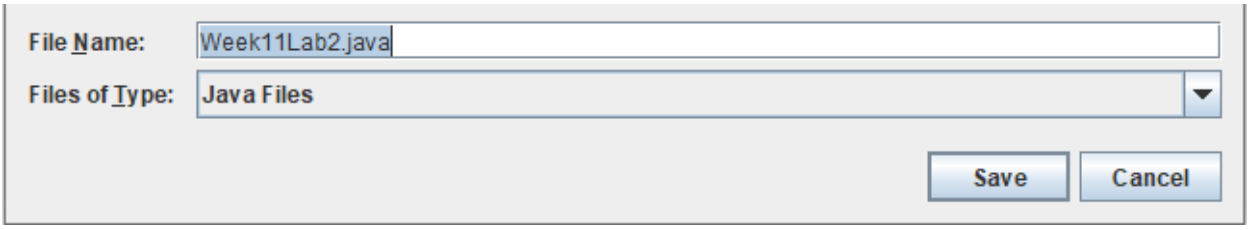
Step 1: Login to your lab computer and create a new java file in jGrasp.

Step 2: Now, type the following code into jGrasp.

```
/* DSA Week 11 Lab 2 */

class BinaryTreeNode {
    String value;
    BinaryTreeNode left;   // First child
    BinaryTreeNode right;  // Next sibling

    BinaryTreeNode(String value) {
        this.value = value;
        this.left = null;
        this.right = null;
    }

    //method to add child using left-child right-sibling nodes
    void addChild(BinaryTreeNode child) {
        if (left == null) {
            left = child;
        } else {
            BinaryTreeNode current = left;
            while (current.right != null) {
                current = current.right;
            }
            current.right = child;
        }
    }
}

public class Week11Lab2 {

    //printTree method for binary tree using left-child right-sibling
    static void printTree(BinaryTreeNode node, int level) {
        if (node == null) return;

        // Indentation loop
        for (int i = 0; i < level; i++) {
            System.out.print("    ");
        }
        System.out.println(node.value);

        // First child (left) goes to next level
        printTree(node.left, level + 1);

        // Sibling (right) stays on same level
        printTree(node.right, level);
    }

```

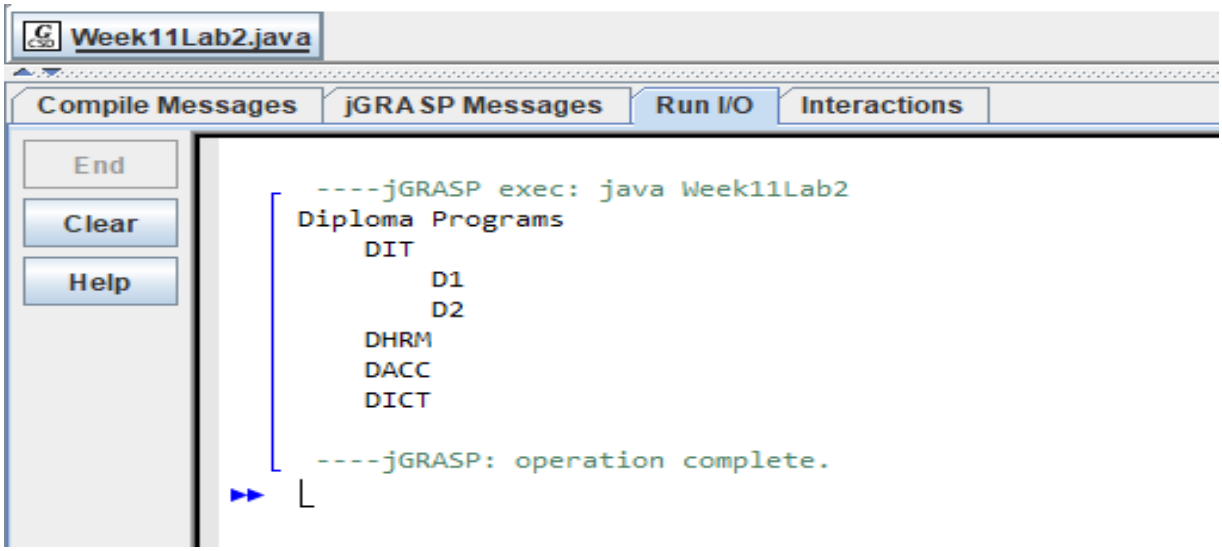
```
47      public static void main(String[] args) {
48          // Create root
49          BinaryTreeNode root = new BinaryTreeNode("Diploma Programs");
50
51          // Add children to root
52          BinaryTreeNode dit = new BinaryTreeNode("DIT");
53          BinaryTreeNode dhrm = new BinaryTreeNode("DHRM");
54          BinaryTreeNode dacc = new BinaryTreeNode("DACC");
55          BinaryTreeNode dict = new BinaryTreeNode("DICT");
56
57          root.addChild(dit);
58          root.addChild(dhrm);
59          root.addChild(dacc);
60          root.addChild(dict);
61
62          // Add children to DIT (first one)
63          dit.addChild(new BinaryTreeNode("D1"));
64          dit.addChild(new BinaryTreeNode("D2"));
65
66          // Print the binary tree
67          printTree(root, 0);
68      }
69  }
70
```

Step 3: After coding the program, go to **file/save** to save your java program as **Week15Lab2**



Step 4: After saving, compile (**click on compile icon or on your keyword hold Ctrl + B**) to check for syntax errors.

Step 5: If compiling is successfully then run (**click on the find and run main method icon or on your keyboard hold Ctrl + R**) your program.



Step 6: If successful your program should display an output like shown in the screenshot above.

Step 7: Week11Lab2 Completed! You have created your first binary tree data structure program.