CS 5007 Individual Project Assignment 6 (100 Points)
Due: 11:59 p.m. on 12/16/2021

Project Objective: The goal of this project is to practice using BinarySearchTree to complete a simple application that can accept **ANY** number of tree node integer values to create a binary tree and output the parent of its nodes, the number of non-leaf nodes, and the number of leaf nodes shown in the sample output.

Project Deliverables:

1. Submit a zip file that contains the **BinarySearchTree.py** and **Node.py** that we discussed in the class and your driver program **binarysearchtree_app.py** file with the **hierarchy chart** to Canvas.
2. Provide the clear **comment documentations** to your code.

Note:

(1) This project is to be done by each student individually. No help besides the textbook, materials, and the instructor should be taken. Copying any answers or part of answers from other sources (including your classmates) will earn you a grade of zero.
(2) All programming conventions mentioned in class should be followed.
(3) You should test your program before submitting.
(4) Your program must be developed and implemented in the PyCharm IDE, or 10% of the graded score is deducted.
(5) Assignments are accepted in their assigned Canvas drop box without penalty if they are received by 11:59PM EST on the due date, or 10% of the graded score is deducted for the late submission per day. Work submitted after one week of its original due date will not be accepted.

Task:

(1) Please extend the given **Node.py** and **BinarySearchTree.py** classes to include the below features:

- Each node itself in a tree can record, access, and update its parent node.
- An instance method returns a node's parent in a tree: getParent().
- An instance method counts the number of leaf nodes in a tree: countLeaves().
- An instance method counts the number of non-leaf nodes in a tree: countNonLeaves()
- An instance method displays the parent node of all its nodes in a tree: printParentNode()

*****You can decide if those methods have the formal parameters, are void methods, or are return methods*****

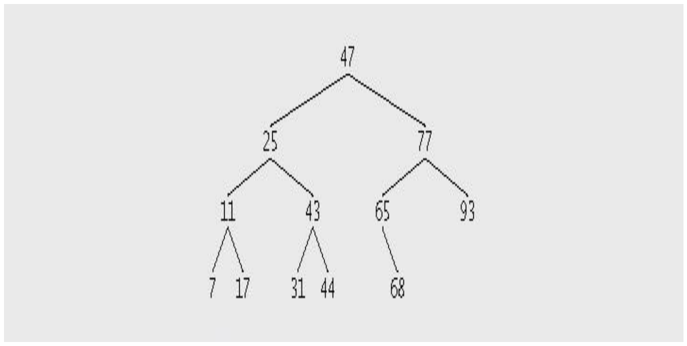
(2) Using the extended **BinarySearchTree.py** in (1), write a driver program, **binarysearchtree_app.py**, to create a binary tree. This program should have three functions: main(), createBinaryTree(), and printBinaryTree().

- The main() function will call the createBinaryTree() function to create a binary tree **ONLY**, using the below sample tree node integer values as an example, and then call the printBinaryTree() function to display the below output that includes the parent node of all its nodes, the number of leaf nodes, and the number of non-leaf nodes. The main() function will also ask the user if there is any new tree to be created. If yes, the main() function will call the createBinaryTree() function to create a new tree again. If no, the main() function will display the message "**Thank You for Using My Application**".

*****You can decide what prompt messages are delivered to the users for inputs*****

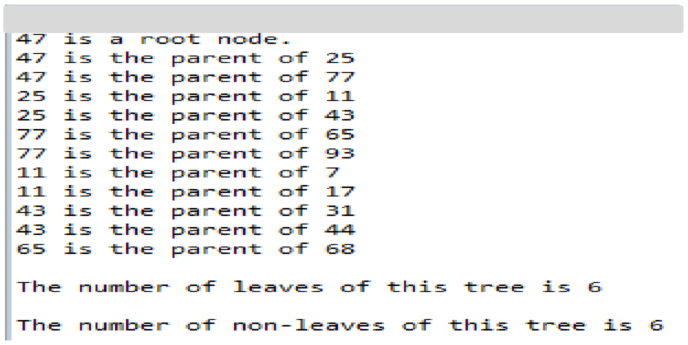
- The createBinaryTree() function will ask the user to input **ONE** node integer value at a time to create a tree **NOT** a sequence of node integer values. It means if the user enters "Yes", the function will ask the user to enter a new node integer value again, or else no more node integer value is allowed to be entered. Then the function returns the created tree to the main() function, which will call the printBinaryTree() function to display the content of that tree shown below. Note that the tree node integer values are not limited to the below sample values. The createBinaryTree() function can create any integer binary tree.

*****You can decide what prompt messages are delivered to the users for inputs*****



*You don't need to display the above tree images by your apps. This tree image is just to show you an example of tree values that you can enter to construct the above tree.

- The printBinaryTree() function will accept the created tree, as the input, from the createBinaryTree() function to print the content of that tree, including the parent node of all its nodes, the number of leaf nodes, and the number of non-leaf nodes shown below.



```
47 is a root node.
47 is the parent of 25
47 is the parent of 77
25 is the parent of 11
25 is the parent of 43
77 is the parent of 65
77 is the parent of 93
11 is the parent of 7
11 is the parent of 17
43 is the parent of 31
43 is the parent of 44
65 is the parent of 68

The number of leaves of this tree is 6

The number of non-leaves of this tree is 6
```

Grading Criteria:

Checkpoint:	**Possible Points**
Hierarchy Chart (.pdf)	5
Proper Naming Conventions	5
Program Documentation	5
Node Class	5
BinarySearchTree Class	15
main()	15
createBinaryTree()	15
printBinaryTree()	15
Proper Prompts for Asking User's Inputs	10
Correct Above Sample Program Outputs	10
Total	100