**CSD203 – Assignment03 - Tree**

**(10 marks – 90 minutes)**

The given file Q2.py already contains statements to implement a simple program using Binary Search Tree (BST) structure. In this test, the BST only contains the nodes with **key<100**. You should write statements to the following functions:

1. (1 mark) Insert(k): insert a new node with key=k to the BST.
2. (1 mark) f1(k): Search number k in the Binary Search Tree.

(Return **True** if this number k exists in the BST, and **False** if it does not exist)

1. (1 mark) f2(): Identify the height of the Binary Search Tree.
2. (1 mark) f3(): Count the amount of leaf nodes in the Binary Search Tree with key>10.
3. (1 mark) f4(): Compute the sum of all leaf nodes of Binary Search Tree.
4. (1 mark) f5(k): Insert an even number to the Binary Search Tree, then return the height of the new BST.
5. (1 mark) f6(): Compute the balance factor of the root.
6. (1 mark) f7(): Count the amount of imbalance nodes.
7. (1 mark) f8(): Find the maximum key.
8. (1 mark) f9(): Delete the minimum node, then compute and return the height of the new tree.

**Notes**:

*- Do not edit given “main” function.*

*- Only perform processing, do not input and output data in any functions.*

- You can create new functions if you see they are necessary.

- You can use <https://www.cs.usfca.edu/~galles/visualization/BST.html> to observe the tree and check the results.

- With each question, you should delete the “pass” statement, then write your code.

**Submit:**

- Compress folder Q2 to Q2.rar or Q2.zip

- Submit file Q2.rar or Q2.zip to lms

----- End -----