# Data Structures and Algorithms Lab

# 10. Trees

**Subject Code:** 17ECSP201 **Lab No:** 10 **Semester:** III

**Date:** Sept, 2018 **Batch:** D

## Question: Computer Representation of a Binary Search Tree

## Objective: Usage of list representation to implement a BST and its operations

---

Create a new project called 'bst' and provide the following functionalities:

- Insert into BST
- Delete from BST
- Traversals: inorder, preorder and postorder

**Once you have the working code, implementing the following functions:**

1. Print the address of root node of the tree.

2. Count the total number of nodes in the tree.

3. Count the number of leaf nodes in the tree.

4. Count and print the number of edges in the tree.

5. Find and delete all the duplicate nodes from the tree.

6. Count the number of nodes having value greater than the given value K.

7. Print the in-order predecessor of the given item.

8. Print the in-order successor of the given item.

9. Find the minimum valued item from the tree.

10. Find the maximum valued item from the tree.

11. Make a duplicate copy of the existing binary search tree and print it.

12. Find and print the number of comparisons made to search a given item from the tree.

13. Count the number of nodes present at level 1 of the tree.

14. Implement the insert_into_bst function using recursion.

15. Count and print the number of internal nodes present in the tree.

16. Find the memory occupied by the tree in terms of bytes.

17. Find the number of edges between root node and the largest element in the tree

18. Print the out-degree of root node.

19. Check if the BST is a strictly binary tree.

20. Implement the recursive Tree search algorithm given below:

```
TREE-SEARCH (x, k)
  If x = NULL or k = key[x]
     then return x
  If k < key[x]
     then return TREE-SEARCH(left[x], k)
    else return TREE-SEARCH(right[x], k)
```

**** Happy Coding ****