SSN College of Engineering, Kalavakkam
Department of Computer Science and Engineering
III Semester - CSE
UCS 1312 Data Structures Lab Laboratory

Academic Year: 2021-2022 **Batch: 2018-2022**

Assignment Date: 08.11.2021

Exercise 6: Binary Search Tree and its Applications

[CO1,K3]

The structure BST has integer data and pointers to left and right children. Implement the following methods.

void insert(struct BST *t, int x) – Insert an integer data into BST
void delete(struct BST *t, int x) – Delete node from BST
void inorder(struct BST *t) – Display the tree using inorder traversal
void levelorder(struct BST *t) – Display the tree using level order traversal
struct BST *find(struct BST *t, int x) – Find the value in x in the tree and return the address of that node
struct BST *findmin(struct BST *t) – Find the minimum in the tree and return the address of that node

Create BSTADTImpl.h with the implementations of the above-mentioned operations
Create BSTADTAppl.c that utilizes BSTADT and BSTADTImpl to perform the operations.

1. Demonstrate the BSTADT with the following test case

   Insert(t,29)
   Insert(t,23)
   Insert(t,4)
   Insert(t,13)
   Insert(t,39)
   Insert(t,31)
   Insert(t,45)
   Insert(t,56)
   Insert(t,49)
   Inorder(t) → 4,13,23,29,31,39,45,49,56
   Levelorder(t) → 1st Level → 29
   2nd level → 23, 39
   3rd Level → 4, 31, 45
   4th Level → 13, 56
   5th Level → 49
   Findmin(t) → 4
   Find(t, 13) → Found, value is 3
   Find(t,3) → Not found

2. Write an application to do the following
   a. Check whether the two BST contains the same set of elements
   b. Check whether the BST is complete or not
   c. Count the number of nodes in tree within the given range

Test case for the Application

(a)

Input: Tree 1

```
        10
      /    \
    5       50
   /       /  \
 1       40   100
```

Input: Tree2

```
        10
      /    \
    5       50
   /       /  \
 1       40   100
```

Tree1 and Tree2 are identical with a set of elements

(b)

Tree1 not complete

(c)

Tree1 Range: [**5**, 45]
    Output: 3
    Nodes are 5, 10, 40

    Tree2 Range: [**1**, 45]
    Output: 4
    Nodes are 1,5, 10, 40