National University of Computer and Emerging Sciences

Laboratory Manual

for

Data Structures Lab

Department of Computer Science FAST-NU, Lahore,

Pakistan

Objectives:

In this lab, students will practice:

1. Binary Search Trees

2. Recursive insert operation, recursive and iterative inorder traversal, and some other recursive operations on BST

Question 1

Implement the following Tree Node:

```
template <typename k, typename v>
struct TNode
{
k key;
v value;
TNode<k, v> *leftChild;
TNode<k, v> *rightChild;
}
```

Now implement a binary search tree class “BST” which contains root of type TNodeasdatamember. You have to implement the following member functions for your binary searchtree:

a. A Copy Constructor , destructor.

b. A function “insert” which is passed as parameter a key and a corresponding value. It then**iteratively** inserts the <key, value> pair while considering the insertion rules. If the keyalreadyexists in the BST, simply replace the value.

```
void insert(k const key, v const value)
```

c. A recursive “insertRec” function which is passed as parameter a key and a correspondingvalue.It then uses **recursion** to insert the <key, value> pair while considering the insertionrules. If thekey already exists in the BST, it simply replaces the value.

```
void insertRec(k const key, v const value)
```

d. A function “inorderTraversal” which prints the keys using **iterative** inorder traversal. `void inorderTraversal() const`

e. A function “NumberOfNodes”. The function then uses **recursion** and count the numberofnodes in a tree. If there is no node, then function returns null.

```
v* NumberOfNodes
```

f. A function “Height” which prints the keys using **recursion** and find height of the givennode. `void Height() const`

g. A function “Leaf” to find the number of the leaf nodes of tree in BST and print the number. `int Leaf() const`

h. Given an array that represents a tree in such a way that array indexes are values in tree nodes and array values give the parent node of that particular index (or node). The value of the root node index would always be -1 as there is no parent for root. Construct the standard linked representation of a given Binary Tree from this given representation.

Case 1: (0—n-1)

```
if (say)father=p;
then left_son=(2*p)+1;
and right_son=(2*p)+2;
```

Case 2: 1—n

```
if (say)father=p;
then left_son=(2*p);
and right_son=(2*p)+1;
```


Question 2: Now run the following main program.

```
int main()
{
	BST<int, int> tree;

	tree.insert(500, 500);
	tree.insertRec(1000, 1000);
	tree.insert(1, 1);
	tree.insert(600, 600);
	tree.insertRec(700, 700);
	tree.insert(10, 10);
	tree.insert(30, 30);
	tree.insertRec(9000, 9000);
	tree.insert(50000, 50000);
	tree.insertRec(20, 20);
	tree.Height(9000,9000);

	cout << "Printing keys using iterative inorder traversal: ";
	tree.inorderPrintKeys();


	system("pause");
}
```