Nama : Moh. Hilman Fariz
Kelas : IF 45 09
NIM : 1301213393
TP MOD 15

Header

```
#ifndef TREE_H_INCLUDED
#define TREE_H_INCLUDED
#include <iostream>

using namespace std;

#define info(t) (t)->info
#define right(t) (t)->right
#define left(t) (t)->left

typedef struct node * adrnode;
typedef int infotype;

struct node{
   infotype info;
   adrnode left, right;
};


adrnode newNode_1301213393(infotype x);
adrnode findNode_1301213393(adrnode root, infotype x);
void insertNode_1301213393(adrnode &root, adrnode p);
void printPreOrder_1301213393(adrnode root);
void printDescendant_1301213393(adrnode root);
int sumNode_1301213393(adrnode root);
int countLeaves_1301213393(adrnode root);
int heightTree_1301213393(adrnode root);

#endif // TREE_H_INCLUDED
```

Tree.cpp

```
#include "tree.h"

adrnode newNode_1301213393(infotype x){
   adrnode emptyNode = new node;
   info(emptyNode) = x;
   right(emptyNode) = NULL;
   left(emptyNode) = NULL;
   return emptyNode;
}

adrnode findNode_1301213393(adrnode root, infotype x) {
   if (root == NULL || info(root) == x) {
      return root;
   } else if (info(root) < x) {
      return findNode_1301213393(right(root), x);
   } else {
      return findNode_1301213393(left(root), x);
   }
}

void insertNode_1301213393(adrnode &root, adrnode p){
   if(root == NULL){
      root = p;
   }else{
      if(info(p) < info(root)){
      if(left(root) == NULL){
         left(root) = p;
      }insertNode_1301213393(left(root), p);
   }
   if(info(p) > info(root)){
      if(right(root) == NULL){
         right(root) = p;
      }else{
         insertNode_1301213393(right(root), p);
         }
      }
   }
}

void printPreOrder_1301213393(adrnode root){
   if(root != NULL){
      cout << info(root) << " ";
      printPreOrder_1301213393(left(root));
      printPreOrder_1301213393(right(root));
   }
}
```

```
void printDescendant_1301213393(adrnode root) {
   if(root != NULL) {
      cout << info(root) << " ";
      printDescendant_1301213393(left(root));
      printDescendant_1301213393(right(root));
   }
}

int sumNode_1301213393(adrnode root){
   if(root == NULL){
      return 0;
   }else{
      return info(root) + sumNode_1301213393(left(root)) + sumNode_1301213393(right(root));
   }
}

int countLeaves_1301213393(adrnode root){
   if(root == NULL){
      return 0;
   }
   if(left(root) == NULL && right(root) == NULL){
      return 1;
   }
   return  countLeaves_1301213393(left(root)) + countLeaves_1301213393(right(root));
}

int heightTree_1301213393(adrnode root){
   int leftHeight;
   int rightHeight;
   if(root == NULL){
      return -1;
   }else{
      leftHeight = heightTree_1301213393(left(root));
      rightHeight = heightTree_1301213393(right(root));
   }
   return max (leftHeight, rightHeight) +1;
}
```

Main.cpp

```
#include "tree.h"
#include <iostream>

using namespace std;

int main()
{
   int x[9] = {5, 3, 9, 10, 4, 7, 1, 8, 6};
   cout << "=======================================================" << endl;
   adrnode root = newNode_1301213393(x[0]);
   for (int i = 0; i < 9; i++) {
      cout << x[i] << " ";
   }
   cout << endl;

   for (int i = 1; i < 9; i++) {
      insertNode_1301213393(root, newNode_1301213393(x[i]));
   }

   cout << endl << "Pre Order\t\t: ";
   printPreOrder_1301213393(root);
   cout << endl << endl;
   cout << "Descendent of Node 9\t: ";
   printDescendant_1301213393(findNode_1301213393(root, 9));
   cout << endl << endl;
   cout << "Sum of BST Info\t\t: " << sumNode_1301213393(root) << endl;
   cout << "Number of Leaves\t: " << countLeaves_1301213393(root) << endl;
   cout << "Height of Tree\t\t: " << heightTree_1301213393(root) << endl;
   cout << "=======================================================" << endl;
   return 0;
}
```

Output