

Deore Mufrad Hendrady
1301223029

Header.h

```
h header.h x C++ main.cpp C++ source.cpp
h header.h > ...
1  #ifndef HEADER_H_INCLUDED
2  #define HEADER_H_INCLUDED
3  #include <iostream>
4
5  using namespace std;
6
7  typedef int infotype;
8
9  typedef struct node *adrNode;
10
11 struct root {
12     adrNode first;
13 };
14
15 struct node {
16     infotype info;
17     adrNode kiri;
18     adrNode kanan;
19 };
20 adrNode newNode_1301223029(infotype x);
21 adrNode findNode_1301223029(adrNode root, infotype x);
22 void insertNode_1301223029(adrNode &root, adrNode p);
23 void printPreOrder_1301223029(adrNode root);
24 void printDescendant_1301223029(adrNode root, infotype x);
25 int sumNode_1301223029(adrNode root);
26 int countLeaves(adrNode root);
27 int heightTree_1301223029(adrNode root);
28 #endif
```

Source.cpp

h header.h

C++ main.cpp

C++ source.cpp X

C++ source.cpp > countLeaves(adrNode)

```
1  #include "header.h"
2  adrNode newNode_1301223029(infotype x){
3      adrNode p;
4      p = new node;
5      p->info= x;
6      p->kanan = NULL;
7      p->kiri = NULL;
8      return p;
9  }
10 adrNode findNode_1301223029(adrNode root, infotype x){
11     if (root == NULL || root ->info == x){
12         return root;
13     }
14     if (x > root->info) {
15         return findNode_1301223029(adrNode root->kanan,x);
16     }else {
17         return findNode_1301223029(root->kiri,x);
18     }
19 }
20
21 void insertNode_1301223029(adrNode &root, adrNode p){
22     if(root == NULL ){
23         root = p;
24     }else {
25         if (p ->info > root->info) {
26             insertNode_1301223029(root->kanan, p);
27         }else {
28             insertNode_1301223029(root->kiri, p);
29         }
30     }
31 }
32 void printPreOrder_1301223029(adrNode root){
33     if (root == NULL){
34         return;
35     }
36
37     cout << root->info << " ";
38     printPreOrder_1301223029(root->kiri );
39     printPreOrder_1301223029(root->kanan);
40
41 }
42 void printDescendant_1301223029(adrNode root, infotype x){
43     adrNode p;
44     p = findNode_1301223029(root , x);
45     printPreOrder_1301223029(p);
46
47 }
48 int sumNode_1301223029(adrNode root){
```

```

C++ source.cpp > countLeaves(adrNode)
39     printPreOrder_1301223029(root->kanan);
40
41 }
42 void printDescendant_1301223029(adrNode root, infotype x){
43     adrNode p;
44     p = findNode_1301223029(root, x);
45     printPreOrder_1301223029(p);
46
47 }
48 int sumNode_1301223029(adrNode root){
49     int hasil = 0;
50     if (root == NULL){
51         return 0;
52     }
53     return root->info + sumNode_1301223029(root->kiri) + sumNode_1301223029(root->ka
54
55
56 }
57 int countLeaves(adrNode root){
58     if (root == NULL){
59         return 0;
60     }
61     if (root->kiri == NULL && root->kanan == NULL){
62         return 1;
63     }else {
64         return countLeaves(root->kiri) + countLeaves(root->kanan);
65     }
66
67 }
68 int heightTree_1301223029(adrNode root){
69     if (root == NULL){
70         return 0;
71     }
72
73     return max(heightTree_1301223029(root->kiri), heightTree_1301223029(root->kanan)
74 }
75

```

main.cpp

```
h header.h  C++ main.cpp  C++ source.cpp
C++ main.cpp > main()
1  #include "header.h"
2
3
4  int main(){
5      int x[9] = {5,3,9,10,4,7,1,8,6};
6
7      adrNode root;
8      root = NULL;
9
10     for (int i = 0 ; i<= 9 -1 ; i++){
11         cout << x[i] << " ";
12         insertNode_1301223029(root, newNode_1301223029(x[i]));
13     }
14
15     cout << endl;
16
17     cout << "Preorder : ";
18     printPreOrder_1301223029(root);
19     cout << endl;
20
21     cout << "descendent of Node 9 : ";
22     printDescendant_1301223029(root,9);
23     cout << endl;
24
25     cout << "Sum of BST Info : " << sumNode_1301223029(root)<<endl;
26
27     cout << "Number of Leaves : " << countLeaves(root)<<endl;
28
29     cout << "Height of Tree : " << heightTree_1301223029(root) << endl;
30     return 0;
31 }
```

output

```
~/STD-TP/TP-15 > on main !1 ?1
./myprogram
5 3 9 10 4 7 1 8 6
Preorder : 5 3 1 4 9 7 6 8 10
descendent of Node 9 : 9 7 6 8 10
Sum of BST Info : 53
Number of Leaves : 5
Height of Tree : 3
```