

```

main.cpp X Bintree.h X Bintree.cpp X
1 #ifndef BINTREE_H_INCLUDED
2 #define BINTREE_H_INCLUDED
3 #include <iostream>
4
5 using namespace std;
6
7 #define left(P) (P->left)
8 #define right(P) (P->right)
9 #define info(P) (P->info)
10
11 typedef int infotype;
12 typedef struct Tree 'adr;
13
14 struct Tree
15 {
16     infotype info;
17     adr left, right;
18 };
19
20 adr createNode_1301213072(infotype x);
21 adr findNode_1301213072(adr root, infotype x);
22 void insertNode_1301213072(adr root, adr P);
23 void printPreOrder_1301213072(adr root);
24 void printDescendant_1301213072(adr root, infotype x);
25 int sumNode_1301213072(adr root);
26 int countLeaves_1301213072(adr root);
27 int heightTree_1301213072(adr root);
28
29 #endif // BINTREE_H_INCLUDED
30

```

```

main.cpp X Bintree.h X Bintree.cpp X
1 #include "Bintree.h"
2
3 adr createNode_1301213072(infotype x)
4 {
5     adr P = new Tree;
6     info(P) = x;
7     left(P) = NULL;
8     right(P) = NULL;
9
10    return P;
11 }
12
13 adr findNode_1301213072(adr root, infotype x)
14 {
15     if (root == NULL) {
16         return NULL;
17     }
18     if (info(root) == x) {
19         return root;
20     }
21     if (info(root) < x) {
22         findNode_1301213072(right(root), x);
23     }
24     else {
25         findNode_1301213072(left(root), x);
26     }
27 }
28
29 void insertNode_1301213072(adr root, adr P)
30 {
31     if (root == NULL) {
32         root = P;
33     }
34     else {
35         if (info(P) < info(root)) {
36             insertNode_1301213072(left(root), P);
37         }
38         else if (info(P) > info(root)) {
39             insertNode_1301213072(right(root), P);
40         }
41         else {
42             cout << "Data yang diinput sudah ada" << endl;
43         }
44     }
45 }
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87

```

```

42 void printPreOrder_1301213072(adr root)
43 {
44     if (root != NULL) {
45         cout << " | " << info(root) << " | ";
46         printPreOrder_1301213072(left(root));
47         printPreOrder_1301213072(right(root));
48     }
49 }
50
51 void printDescendant_1301213072(adr root, infotype x)
52 {
53     adr P = findNode_1301213072(root, x);
54
55     if (P == NULL) {
56         cout << "Data node tidak ada" << endl;
57     }
58     else {
59         printPreOrder_1301213072(left(P));
60         printPreOrder_1301213072(right(P));
61     }
62 }
63
64 int sumNode_1301213072(adr root)
65 {
66     if (root == NULL) {
67         return 0;
68     }
69     else {
70         return info(root) + sumNode_1301213072(left(root)) + sumNode_1301213072(right(root));
71     }
72 }
73
74 int countLeaves_1301213072(adr root)
75 {
76     if (root == NULL) {
77         return 0;
78     }
79     else if (left(root) == NULL && right(root) == NULL) {
80         return 1;
81     }
82     else {
83         return countLeaves_1301213072(left(root)) + countLeaves_1301213072(right(root));
84     }
85 }
86
87 int heightTree_1301213072(adr root)
88 {
89     if (root == NULL) {
90         return -1;
91     }
92     else {
93         return (heightTree_1301213072(left(root)), heightTree_1301213072(right(root))) + 1;
94     }
95 }
96
97
98
99
100

```

```
main.cpp x Bintree.h x Bintree.cpp x
1 //include "Bintree.h"
2
3 int main()
4 {
5     cout << "++++++" << endl << endl;
6
7     int x[9] = {5, 3, 9, 10, 4, 7, 1, 8, 6};
8
9     for(int i = 0; i < 9; i++)
10     {
11         cout << x[i] << " ";
12     }
13
14     cout << endl;
15
16     adr root = NULL;
17
18     for(int i = 0; i < 9; i++)
19     {
20         adr P = createNode_1301213072(x[i]);
21         insertNode_1301213072(root, P);
22     }
23
24     printf("\n");
25     printf("\nPre Order\t: ");
26     printPreOrder_1301213072(root);
27
28     printf("\n");
29     printf("\nDescendent of Node 9\t: ");
30     printDescendant_1301213072(root, 9);
31
32     printf("\n");
33     printf("\nSum of BST Info\t: ");
34     cout << sumNode_1301213072(root);
35
36     printf("\nNumber of Leaves\t: ");
37     cout << countLeaves_1301213072(root);
38
39     printf("\nHeight of Tree\t: ");
40     cout << heightTree_1301213072(root);
41
42     cout << endl << endl << "++++++" << endl;
43     return 0;
44 }
45
```

++++++

5 3 9 10 4 7 1 8 6

Pre Order : || 5 || || 3 || || 1 || || 4 || || 9 || || 7 || || 6 || || 8 || || 10 ||

Descendent of Node 9 : || 7 || || 6 || || 8 || || 10 ||

Sum of BST Info : 53

Number of Leaves : 5

Height of Tree : 2

++++++

Process returned 0 (0x0) execution time : 0.050 s

Press any key to continue.