## **Duta Razaq Suhoyo**

#### 1301210280

#### IF 45 11

#### Tree.h

```
Tree.h X Tree.cpp X main.cpp X
          #ifndef TREE H INCLUDED
     1
          #define TREE H INCLUDED
     2
     3
     4
          #include <iostream>
     5
          using namespace std;
     6
     7
          #define info(p) (p)->info
     8
          #define right(p) (p)->right
     9
          #define left(p) (p)->left
          #define nil NULL
    10
    11
    12
          typedef int infotype;
    13
          typedef struct node *adrnode;
    14
    15 ⊟struct node{
              infotype info;
    16
    17
              adrnode right;
    18
              adrnode left;
    19
         L};
    20
    21
          adrnode newnode 1301210280(infotype x);
          adrnode findnode 1301210280 (adrnode root, infotype x);
    22
    23
          void insertnode 1301210280 (adrnode &root, adrnode p);
    24
          void printpreorder 1301210280 (adrnode root);
          void printdescendant 1301210280(adrnode root, infotype x);
    25
    26
          int sumnode 1301210280 (adrnode root);
    27
          int countleaves 1301210280(adrnode root);
    28
          int heightTree 1301210280(adrnode root);
    29
    30
          #endif // TREE H INCLUDED
    31
```

#### Tree.cpp

```
Tree.h X Tree.cpp X main.cpp X
          #include "Tree.h"
     2
     3
       □adrnode newnode 1301210280(infotype x){
              adrnode p = new node;
     4
     5
              info(p) = x;
     6
              right(p) = nil;
     7
              left(p) = nil;
         L
     8
     9
    10
       □adrnode findnode 1301210280 (adrnode root, infotype x) {
    11 📋
              if(root == nil){
    12
                  return nil;
              }else if(info(root) == x){
    13
    14
                  return root;
    15
              }else if(info(root) > x){
    16
                  return findnode 1301210280(left(root), x);
    17
    18
                  return findnode 1301210280(right(root), x);
    19
         L
    20
    21
        □void insertnode 1301210280 (adrnode &root, adrnode p) {
    22
    23 🖹
              if(root == nil){
    24
                  root = p;
    25
              }else if(info(p) < info(root)){</pre>
    26
                  insertnode_1301210280(left(root), p);
    27
              }else{
    28
                  insertnode 1301210280(right(root), p);
    29
         L
    30
    31
    32
        □void printpreorder 1301210280 (adrnode root) {
              if(root != nil){
    33
    34
                  cout << info(root) << " ";</pre>
    35
                  printpreorder 1301210280(left(root));
    36
                  printpreorder 1301210280(right(root));
    37
              }
    20
```

```
Tree.h X Tree.cpp X main.cpp X
   39
    40
       □void printdescendant_1301210280(adrnode root, infotype x) {
    41
           if(root != nil) {
               printdescendant_1301210280(left(root), x);
   42
   43
                if(info(root) != x){
    44
                     cout << info(root) << " ";</pre>
   4.5
                printdescendant_1301210280(right(root), x);
    46
    47
   48
    49
    50
        pint sumnode_1301210280 (adrnode root) {
       if(root == nil){
    51
                return 0;
   52
    53
                 return info(root) + sumnode_1301210280(left(root)) + sumnode_1301210280(right(root));
    54
   55
    56
    57
    58 pint countleaves_1301210280(adrnode root){
       if (root == nil) {
                return 0;
   60
             }else if(left(root) == nil && right(root) == nil){
    61
    62
   63
             }else{
                 return countleaves_1301210280(left(root)) + countleaves_1301210280(right(root));
    64
    65
    66
    67
        □int heightTree 1301210280 (adrnode root) {
    68
    69 = if(root == nil){
   70
                return -1;
   71
             }else{
   72
                return max(heightTree 1301210280(left(root))+1, heightTree 1301210280(right(root))+1);
   73
   74
   75
```

#### Main.cpp

```
Tree.h X Tree.cpp X main.cpp X
           #include <iostream>
#include "Tree.h"
           using namespace std;
           int main()
     8
                cout<<" =======
                                             int x[9] = \{5, 3, 9, 10, 4, 7, 1, 8, 6\};
     9
    10
                int i = 0;
    11
                while(i<9) {
    cout<<" "<<x[i];</pre>
    12
    13
    14
                    i++;
    15
    16
    17
                i = 0;
    18
                adrnode p;
    19
                adrnode root = nil;
    20
    21
                while(i<9){
                    p = newnode_1301210280(x[i]);
    22
                     insertnode_1301210280(root, p);
    23
    24
25
                    i++;
    26
               printf(" \n ");
printf(" \n Pre Order\t\t: ");
printpreorder_1301210280(root);
    27
    28
    29
    30
                printf(" \n ");
printf(" \n Descendent of Node 9\t: ");
    31
    32
    33
                p = findnode 1301210280(root, 9);
    34
                printdescendant_1301210280(p,9);
    35
                printf(" \n ");
printf(" \n Sum of BST Info\t: ");
    36
    37
        Tree.cpp X main.cpp X
Tree.h X
                while(i<9) {
    cout<<" "<<x[i];
    12
    13
    14
    15
    16
    17
               i = 0;
    18
                adrnode p;
    19
                adrnode root = nil;
    20
    21
                while(i<9){
                    p = newnode 1301210280(x[i]);
    22
    23
                     insertnode_1301210280(root, p);
    24
    25
    26
               printf(" \n ");
printf(" \n Pre Order\t\t: ");
    27
28
29
               printpreorder_1301210280(root);
    30
               printf(" \n ");
printf(" \n Descendent of Node 9\t: ");
    31
    32
    33
                p = findnode 1301210280(root, 9);
    34
                printdescendant_1301210280(p,9);
    35
                printf(" \n ");
printf(" \n Sum of BST Info\t: ");
    36
    37
                cout<<sumnode_1301210280(root);</pre>
    38
39
                printf(" \n Number of Leaves\t: ");
    40
    41
                cout<<countleaves_1301210280(root);</pre>
    42
    43
                printf(" \n Height of Tree\t\t: ");
     44
                cout<<heightTree_1301210280(root)<<endl;</pre>
    45
                cout<<" =======
    46
    47
    48
```

### Output

# ■ "D:\Telkom Univ\SEM 3\PRAK STRUKDAT\TP MOD15\TP MOD15\bin\Debug\TP MOD15.exe"

\_\_\_\_\_\_

5 3 9 10 4 7 1 8 6

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

Descendent of Node 9 : 6 7 8 10

Sum of BST Info : 53 Number of Leaves : 5 Height of Tree : 3

\_\_\_\_\_\_\_