TDM-GCC 4.9.2 64-bit Release

(globals)

Project   Classes   Debug   [*] tree.cpp

```c
#include <stdio.h>
#include <stdlib.h>

struct node {
    int data;

    struct node *leftChild;
    struct node *rightChild;
};

struct node *root = NULL;

void insert(int data) {
    struct node *tempNode = (struct node*) malloc(sizeof(struct node));
    struct node *current;
    struct node *parent;

    tempNode->data = data;
    tempNode->leftChild = NULL;
    tempNode->rightChild = NULL;


    if(root == NULL) {
        root = tempNode;
    } else {
        current = root;
        parent = NULL;

        while(1) {
            parent = current;


            if(data < parent->data) {
                current = current->leftChild;


                if(current == NULL) {
                    parent->leftChild = tempNode;
                    return;
                }
```

ENG
IN

17:50
28-09-2022

```cpp
16          struct node *parent;
17
18          tempNode->data = data;
19          tempNode->leftChild = NULL;
20          tempNode->rightChild = NULL;
21
22
23      if(root == NULL) {
24          root = tempNode;
25      } else {
26          current = root;
27          parent = NULL;
28
29          while(1) {
30              parent = current;
31
32
33              if(data < parent->data) {
34                  current = current->leftChild;
35
36
37                  if(current == NULL) {
38                      parent->leftChild = tempNode;
39                      return;
40                  }
41              }
42              else {
43                  current = current->rightChild;
44
45
46                  if(current == NULL) {
47                      parent->rightChild = tempNode;
48                      return;
49                  }
50              }
51          }
52      }
```

```cpp
67
68          else {
69              current = current->rightChild;
70          }
71
72          //not found
73          if(current == NULL) {
74              return NULL;
75          }
76      }
77
78      return current;
79  }
80
81  void pre_order_traversal(struct node* root) {
82      if(root != NULL) {
83          printf("%d ",root->data);
84          pre_order_traversal(root->leftChild);
85          pre_order_traversal(root->rightChild);
86      }
87  }
88
89  void inorder_traversal(struct node* root) {
90      if(root != NULL) {
91          inorder_traversal(root->leftChild);
92          printf("%d ",root->data);
93          inorder_traversal(root->rightChild);
94      }
95  }
96
97  void post_order_traversal(struct node* root) {
98      if(root != NULL) {
99          post_order_traversal(root->leftChild);
100         post_order_traversal(root->rightChild);
101         printf("%d ", root->data);
102     }
103 }
```

ENG
IN

17:50
28-09-2022

TDM-GCC 4.9.2 64-bit Release

(globals)

Project   Classes   Debug   [*] tree.cpp

```cpp
107        int i;
108        int array[7] = { 27, 14, 35, 10, 19, 31, 42 };
109
110        for(i = 0; i < 7; i++)
111            insert(array[i]);
112
113        i = 31;
114        struct node * temp = search(i);
115
116        if(temp != NULL) {
117            printf("[%d] Element found.", temp->data);
118            printf("\n");
119        }else {
120            printf("[ x ] Element not found (%d).\n", i);
121        }
122
123        i = 15;
124        temp = search(i);
125
126        if(temp != NULL) {
127            printf("[%d] Element found.", temp->data);
128            printf("\n");
129        }else {
130            printf("[ x ] Element not found (%d).\n", i);
131        }
132
133        printf("\nPreorder traversal: ");
134        pre_order_traversal(root);
135
136        printf("\nInorder traversal: ");
137        inorder_traversal(root);
138
139        printf("\nPost order traversal: ");
140        post_order_traversal(root);
141
142        return 0;
143    }
```

```
1

enter a value to insert into hash table
3

Press 1. Insert  2. Display    3. Search      4.Exit
3

enter search element
4

 value is not found

Press 1. Insert  2. Display    3. Search      4.Exit
2

elements in the hash table are

at index 0       value =  0
at index 1       value =  0
at index 2       value =  0
at index 3       value =  3
at index 4       value =  0
at index 5       value =  0
at index 6       value =  0
at index 7       value =  0
at index 8       value =  0
at index 9       value =  0
Press 1. Insert  2. Display    3. Search      4.Exit
```