

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  typedef struct treeNode{
6      struct treeNode *left;
7      int data;
8      struct treeNode *right;
9  }TreeNode, *TreeNodePtr;
10
11 void inOrder(TreeNodePtr treePtr);
12 void preOrder(TreeNodePtr treePtr);
13 void postOrder(TreeNodePtr treePtr);
14
15 void inOrder(TreeNodePtr treePtr){
16     if(treePtr != NULL){
17         inOrder(treePtr->left);
18         printf("%3d", treePtr->data);
19         inOrder(treePtr->right);
20     }
21 }
22
23 void preOrder(TreeNodePtr treePtr){
24     if(treePtr != NULL){
25         printf("%3d", treePtr->data);
26         preOrder(treePtr->left);
27         preOrder(treePtr->right);
28     }
29 }
30
31 void postOrder(TreeNodePtr treePtr){
32     if(treePtr != NULL){
33         postOrder(treePtr->left);
34         postOrder(treePtr->right);
35         printf("%3d", treePtr->data);
36     }
37 }
38
39 void insertNode(TreeNodePtr *treePtr, int value){
40     if(*treePtr == NULL){
41         *treePtr = malloc( sizeof(TreeNode) );
42         (*treePtr)->data = value;
```

```
43     (*treePtr)->left = NULL;
44     (*treePtr)->right = NULL;
45 }
46 else{
47     if( value < (*treePtr)->data){
48         insertNode( &((*treePtr)->left) ,value);
49     }
50     else if( value > (*treePtr)->data){
51         insertNode( &((*treePtr)->right) ,value);
52     }
53     else{
54         printf("DUP");
55     }
56 }
57 }
58
59 int main(void){
60     TreeNodePtr root = NULL;
61     int index, item;
62
63     srand(time(NULL));
64     printf("Adicionando valores na arvore\n");
65     for(index = 0; index <= 15; index++){
66         item = rand() % 15;
67         printf("%3d", item);
68         insertNode(&root, item);
69     }
70
71     printf("\n\nAtravessando a arvore em Pre Ordem\n");
72     preOrder(root);
73
74     printf("\n\nAtravessando a arvore em In Ordem\n");
75     inOrder(root);
76
77     printf("\n\nAtravessando a arvore em Pos Ordem\n");
78     postOrder(root);
79 }
```