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



struct node{
   int data;
   struct node *right_child;
   struct node *left_child;
};

struct node *new_node(int x){
   struct node* temp;
   temp=(struct node*)malloc(sizeof(struct node));
   temp->data=x;
   temp->right_child=NULL;
   temp->left_child=NULL;
   return temp;
};

struct node *insert (struct node *root, int x){
   if(root==NULL){
       return new_node(x);
   }
   else if(x > root->data){
       root->right_child=insert(root->right_child,x);
   }
   else{
      root->left_child=insert(root->left_child,x) ;
   }
     return root;
};

void preorder(struct node *root){
   if (root!=NULL){
   printf("%d\n", root->data);
   preorder(root->left_child);
   preorder(root->right_child);
   }
};
```

```
void inorder(struct node *root){
    if (root!=NULL){
    inorder(root->left_child);
    printf("%d\n", root->data);
    inorder(root->right_child);
}}

void postorder(struct node *root){
    if (root!=NULL){
    postorder(root->left_child);
    postorder(root->right_child);
    printf("%d\n", root->data);
    }
};



int main(){
    struct node *root=new_node(100);
insert(root, 5);
insert(root, 15);
insert(root, 2);
insert(root, 4);
insert(root, 30);
insert(root, 7);
insert(root, 1);

printf("preorder traversal\n");
preorder(root);
printf("\n");
printf("inorder traversal\n");
inorder(root);
printf("\n");
printf("postorder traversal\n");
postorder(root);
printf("\n");
}
```

Output:

```
preorder traversal
100
5
2
1
4
15
7
30

inorder traversal
1
2
4
5
7
15
30
100

postorder traversal
1
4
2
7
30
15
5
100


Process returned 0 (0x0)   execution time : 0.089 s
Press any key to continue.
```