Binary

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

struct node {
    int key;
    struct node * left, *right;
};

struct node * newNode (int item)
{
    struct node * temp = (struct node *) malloc (sizeof
                                    (struct node));
    temp -> key = item;
    temp -> left = temp -> right = NULL;
    return temp;
}

void inorder ( struct node* root)
{ if (root != NULL) {
        inorder (root -> left);
        printf ("%d \t", root -> key);
        inorder (root -> right);

    }
    if (root == NULL)
    {
        printf ("Tree is empty \n");
        return;
    }
}
```

```
void preorder (struct node *root);
{ if (root != NULL) {
    printf ("%d \t", root->key);
    preorder (root->left);
    preorder (root->right);
}
if (root == NULL)
{ printf ("Tree is empty \n");
  return;
  }
}
void postorder (struct node* root)
{
    if (root != NULL) {
      return
    post order - (root->left);
    postorder (root -> right);
    printf ("%d \t", root->key);
}
}
```

```
struct node * insert (struct node * node, int key)
{ if (node == NULL)
      return newNode(key);
  if (key < node -> key)
      node -> left = insert(node -> left, key);
  else if (key > node -> key)
      node -> right = insert (node -> right, key);
  return node;
  }.
int main( )
{
int ch;
int item;
struct node * root = NULL;
for (;;)
{
printf ("1: Create tree \n2: Insert \n3: InOrder \n4:
        preorder \n 5: Postorder \n");
printf("Enter your choice:");
scanf ("%d", &ch);
switch (ch)
{
case 1 : printf ("Enter the name :");
         scanf ("%d", &item);
     root = insert (root, item);
     break;
```

```
        case 2 : printf ("Enter the item: ");
                 scanf ("%d", &item);
                 insert(root, item);
                 break;
        case 3 : inorder(root);
                 break;
        case 4 : preorder(root);
                 break;
        case 5 : postorder(root);
                 break;
        default : break;
        }
    }
    return 0;
}
```