Write a program

a) To construct a binary serach tree.
b) To traverse the tree using all the methods i.e inorder, preorder & postorder
c) To display the elements in the tree.

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

struct node {
    struct node *llink;
    struct node *rlink;
    int info;
};
typedef struct node *NODE;

NODE getnode()
{ NODE x = (NODE) malloc(sizeof(struct node));
  if(x==NULL){
    printf("Memory full\n");
    exit(0);
  }
  return x;
}
NODE insert(NODE root, int item)
{ NODE temp, curr, prev;
  if(root==NU
    temp->info = item;
```

```
temp->llink = temp->rlink = NULL;
if(root==NULL)
return temp;
prev = NULL;
curr = root;
while( curr != NULL)
{  prev = curr;
   curr = (item < curr->info) ? curr->llink :
                                curr->rlink;
}
if (item < prev->info)
   prev->llink = temp;
else
prev->rlink = temp;
return root;
}
void preorder (NODE root){
  if(root!=NULL){
  printf("%d", root->info);
  preorder(root->llink);
  preorder(root->rlink);
}}
void postorder(NODE root){
  if(root!=NULL){
  printf("%d", root->info);
     postorder(root->llink);
     postorder(root->rlink);
   printf("%d", root->info);
}}
void inorder(NODE root){
  if(root!=NULL){
     postorder(
```

```
    inorder (root->llink);
    printf("%d", root->info);
    inorder (root->rlink);
}
void display (NODE root, int i)
{ int j;
  if (root != NULL)
  {
    display (root->rlink, i+1);
    for (j=0; j<i; j++)
      printf("  ");
    printf("%d\n", root->info);
    display (root->llink, i+1);
  }
  else printf("tree is empty.\n");
}
int main(){
int item, choice;
NODE root = NULL;
for(;;)
{ printf("\n1. insert \t2. display \t3. postorder\t
      4. postorder \t5. inorder \t6. exist\n");
printf("enter the choice\n");
scanf("%d", &choice);
switch(choice)
{
  case 1: printf("enter the item\n");
    scanf("%d", &item); root = insert(root, item);
    break;
  case 2: display(root, 0);
    break; if(root == NULL) printf("Tree is
    empty\n"); else
    else display(root, 0);
```

```
        return; break;
case 3: preorder(root);
        break;
case 4: postorder(root);
        break;
case 5: inorder(root);
        break;
default: printf("wrong choice. THANK YOU...");
         exit(1);
         break;
    }
}
```