10
```
#include<stdio.h>
#include<stdlib.h>
struct node
  { int data;
    struct node *left;
    struct node *right;
  }*root;
struct node *create()
  { struct node *temp;
    printf("Enter the root node element");
    temp=(struct node*)malloc(sizeof(struct node));
    scanf("%d",&temp->data);
    temp->left=temp->right=NULL;
    return temp;
  }
void insert(struct node *root, struct node *temp)
  {
    if(temp->data<root->data)
      { if(root->left!=NULL)
          insert(root->left,temp);
        else
          root->left=temp;
      }
    if(temp->data>root->data)
      { if(root->right!=NULL)
          insert(root->right,temp);
        else
          root->right=temp;
      }
  }
void printPostorder(struct node *node)
  {
```

```
if(node==NULL)
  return;
printpostorder(node->left);
printPostorder(node->right);
printf("%d\t", node->data);
}
void printInorder(struct node*node)
{
  if(node==NULL)
    return;
  printInorder(node->left);
  printf("%d\t", node->data);
  printfInorder(node->right);
}
void printPreorder(struct node* node)
{
  if(node==NULL)
    return;
  printf("%d\t", node->data);
  printPreorder(node->left);
  printPostorder(node, right);
}
int main()
{ int choice;
  struct choice;
  struct node* temp;
  do{
    printf("1.Create\n2.Insert\n3.Preorder\n
      4.Inorder\n5.Postorder\n6.Exit");
    scanf("%d", &choice);
    switch(choice)
      { case 1: root1 = create()
          break;
```

```
case2: printf("Enter the value to insert");
       temp=(struct node*)malloc(sizeof(struct node));
       scanf("%d", &temp->data);
       insert(root1, temp);
       break;
case 3: printPreorder(root1);
        break;
case 4: printInorder(root1);
        break;
case 5: printPostorder(root1);
        break;
case 6: exit(0);
        break;
default: printf("Incorrect choice\n");
}while(choice!=6);
return 0;
}
```