S.No: 21	Exp. Name: ***Write the code for traversing a binary tree in preorder, inorder and postorder***	Date:2023-06-23

Aim:

Write a recursive C program for traversing a binary tree in preorder, inorder and postorder.

Source Code:

binaryTree.c

```
#include<stdio.h>
#include<stdlib.h>
struct node *create(int);
struct node *insert(struct node *,int);
void inorder(struct node *);
void postorder(struct node *);
void preorder(struct node *);
int main()
{
   struct node *root;
   int x,ch;
   while(1)
   {
      printf("0.create\n1.insert\n2.preorder\n3.postorder\n4.inorder\n5.exit\n");
      printf("Enter your choice: ");
      scanf("%d",&ch);
      switch(ch)
      {
         case 0 :
                 printf("Enter the data: ");
                 scanf("%d",&x);
                 root=create(x);
                 break;
         case 1 :
                 printf("Enter the data: ");
                 scanf("%d",&x);
                 root=insert(root,x);
                 break;
         case 2 :
                printf("Display tree in Preorder ");
                preorder(root);
                printf("\n");
                break;
         case 3 :
                printf("Display tree in Postorder ");
                 postorder(root);
                 printf("\n");
                 break;
         case 4 :
                printf("Display tree in Inorder ");
                inorder(root);
                printf("\n");
                break;
         case 5 :
                exit(0);
```


ID: 22K61A05H2

```
                break;
        default:
                printf("Entervalid input\n");
  }
 }
}
struct node
{
   struct node *lchild;
   int info;
   struct node *rchild;
};
struct node *create(int x)
{
   struct node *temp;
   temp=(struct node*)malloc(sizeof(int));
   if(temp==NULL)
   {
      printf("Memory is not allocated \n");
      return 0;
 }
   else
   {
      temp->lchild=NULL;
      temp->info=x;
      temp->rchild=NULL;
 }
}
void inorder(struct node *root)
{
   if(root==NULL)
   {
 }
   else
   {
      inorder(root->lchild);
      printf("%d->",root->info);
      inorder(root->rchild);
 }
}
void preorder(struct node *root)
{
   if(root==NULL)
   {
 }
   else
   {
      printf("%d->",root->info);
      preorder(root->lchild);
      preorder(root->rchild);
 }
}
void postorder(struct node *root)
{
   if(root==NULL)
   {
```

```
 }
   else
   {
      postorder(root->lchild);
      postorder(root->rchild);
      printf("%d->",root->info);
 }
}
struct node *insert(struct node *root,int x)
{
   if(root==NULL)
   {
      return create(x);
 }
   else if(x<root->info)
   {
      root->lchild=insert(root->lchild,x);
 }
   else
   {
      root->rchild=insert(root->rchild,x);
 }
}
```


Execution Results - All test cases have succeeded!

Test Case - 1
User Output
0.create 0
1.insert 0
2.preorder 0
3.postorder 0
4.inorder 0
5.exit 0
Enter your choice: 0
Enter the data: 25
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice: 1
Enter the data: 245
0.create 0
1.insert 0
2.preorder 0
3.postorder 0
4.inorder 0
5.exit 0
Enter your choice: 0
Enter the data: 345

```
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  36
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  589
0.create 2
1.insert 2
2.preorder 2
3.postorder 2
4.inorder 2
5.exit 2
Enter your choice:  2
Display tree in Preorder 345->36->589-> 3
0.create 3
1.insert 3
2.preorder 3
3.postorder 3
4.inorder 3
5.exit 3
Enter your choice:  3
Display tree in Postorder 36->589->345-> 4
0.create 4
1.insert 4
2.preorder 4
3.postorder 4
4.inorder 4
5.exit 4
Enter your choice:  4
Display tree in Inorder 36->345->589-> 5
0.create 5
1.insert 5
2.preorder 5
3.postorder 5
4.inorder 5
5.exit 5
Enter your choice:  5
```

Test Case - 2
User Output

```
0.create 0
1.insert 0
2.preorder 0
3.postorder 0
4.inorder 0
5.exit 0
Enter your choice:  0
Enter the data:  21
0.create 0
1.insert 0
2.preorder 0
3.postorder 0
4.inorder 0
5.exit 0
Enter your choice:  0
Enter the data:  325
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  586
0.create 0
1.insert 0
2.preorder 0
3.postorder 0
4.inorder 0
5.exit 0
Enter your choice:  0
Enter the data:  26
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  478
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  213
0.create 1
1.insert 1
2.preorder 1
```

```
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  36
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  21
0.create 1
1.insert 1
2.preorder 1
3.postorder 1
4.inorder 1
5.exit 1
Enter your choice:  1
Enter the data:  2245
0.create 2
1.insert 2
2.preorder 2
3.postorder 2
4.inorder 2
5.exit 2
Enter your choice:  2
Display tree in Preorder 26->21->478->213->36->2245-> 3
0.create 3
1.insert 3
2.preorder 3
3.postorder 3
4.inorder 3
5.exit 3
Enter your choice:  3
Display tree in Postorder 21->36->213->2245->478->26-> 4
0.create 4
1.insert 4
2.preorder 4
3.postorder 4
4.inorder 4
5.exit 4
Enter your choice:  4
Display tree in Inorder 21->26->36->213->478->2245-> 5
0.create 5
1.insert 5
2.preorder 5
3.postorder 5
4.inorder 5
5.exit 5
```

```
Enter your choice:  5
```