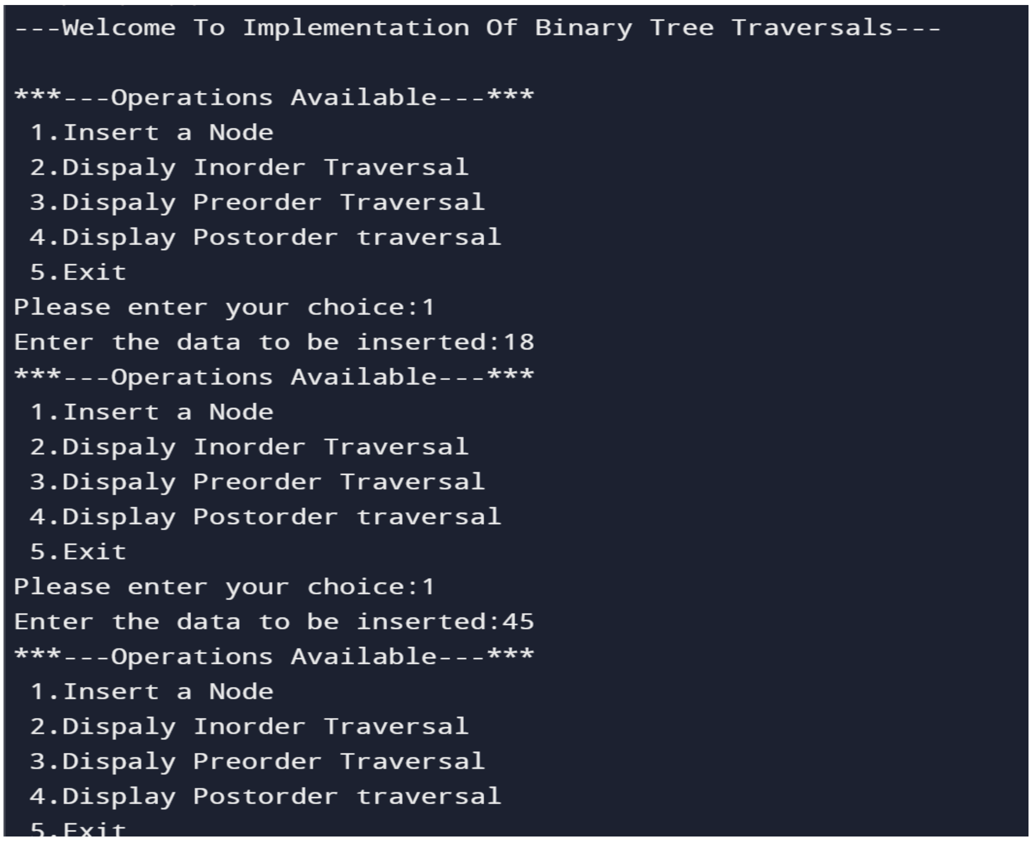
Experiment 6

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

struct node
{
   int data;
   struct node*left;
   struct node*right;
};
struct node*tree;
void create(struct node*);
struct node *insert(struct node *, int);
void inorder(struct node*);
void preorder(struct node*);
void postorder(struct node*);

int choice,x;
struct node*ptr;
void main()
{
   printf("\n---Welcome To Implementation Of Binary Tree Traversals---\n");
   create(tree);
   do
   {
      printf("\n***---Operations Available---***");
      printf("\n 1.Insert a Node");
      printf("\n 2.Dispaly Inorder Traversal");
      printf("\n 3.Dispaly Preorder Traversal");
      printf("\n 4.Display Postorder traversal");
printf("\n 5.Exit \n");
printf("Please enter your choice:");
scanf("%d",&choice);

switch(choice)
{
   case 1:
   printf("\n Enter the data to be inserted:");
   scanf("%d",&x);
   tree = insert(tree,x);
   break;

   case 2:
   printf("\n Elements in the inorder traversal are: ");
   inorder(tree);
   printf("\n");
   break;

   case 3:
   printf("\n Elements in the preorder traversal are: ");
   preorder(tree);
   printf("\n");
   break;

   case 4:
   printf("\n Elements in the postorder traversal are: ");
   postorder(tree);
   printf("\n");
   break;

   case 5:
   printf("Exit : Program Finished");
```

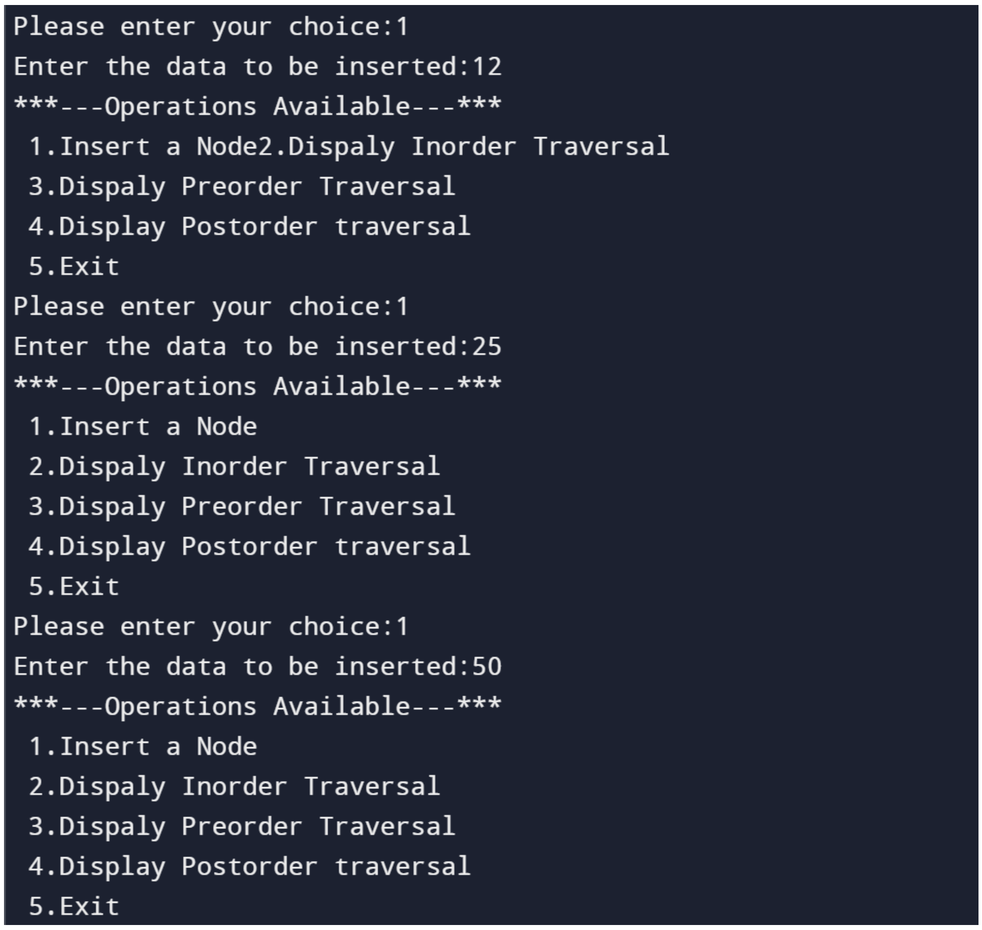
```
            break;

            default :
            printf("\n Please enter a valid option 1,2,3,4,5");
            break;
        }
    }while(choice!=5);
}
void create(struct node*tree)
{
    tree=NULL;
}

struct node*insert(struct node*tree,int x)
{
    struct node*p,*temp,*root;
    p=(struct node *)malloc(sizeof(struct node));
    p->data=x;
    p->left=NULL;
    p->right=NULL;
    if(tree==NULL)
    {
        tree=p;
        tree->left=NULL;
        tree->right=NULL;
    }
    else
    {
        root=NULL;
        temp=tree;
        while(temp!=NULL)
        {
            root=temp;
            if(x<temp->data)
               temp=temp->left;
            else
            temp=temp->right;
        }
        if(x<root->data)
           root->left=p;
        else
        root->right=p;
    }
    return tree;
}

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

void preorder(struct node*tree)
{
    if(tree!=NULL)
    {
        printf("%d\t",tree->data);
        preorder(tree->left);
        preorder(tree->right);
    }
```

```
}

void postorder(struct node*tree)
{
   if(tree!=NULL)
   {
      postorder(tree->left);
      postorder(tree->right);
      printf("%d\t",tree->data);
   }
}
```

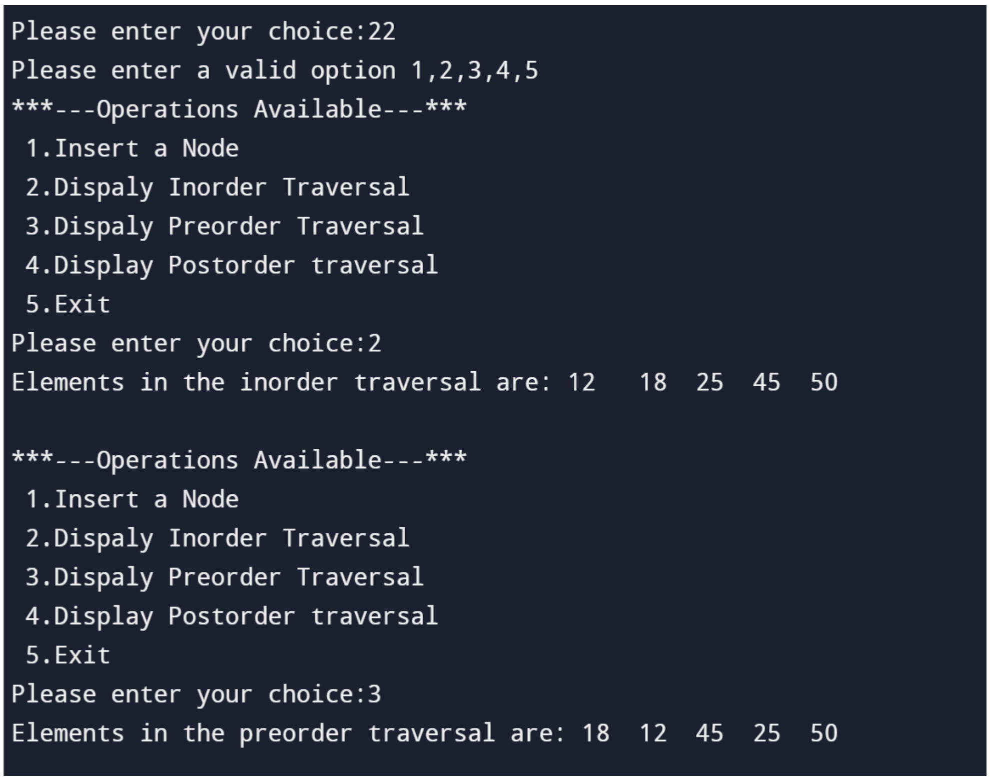
Output:



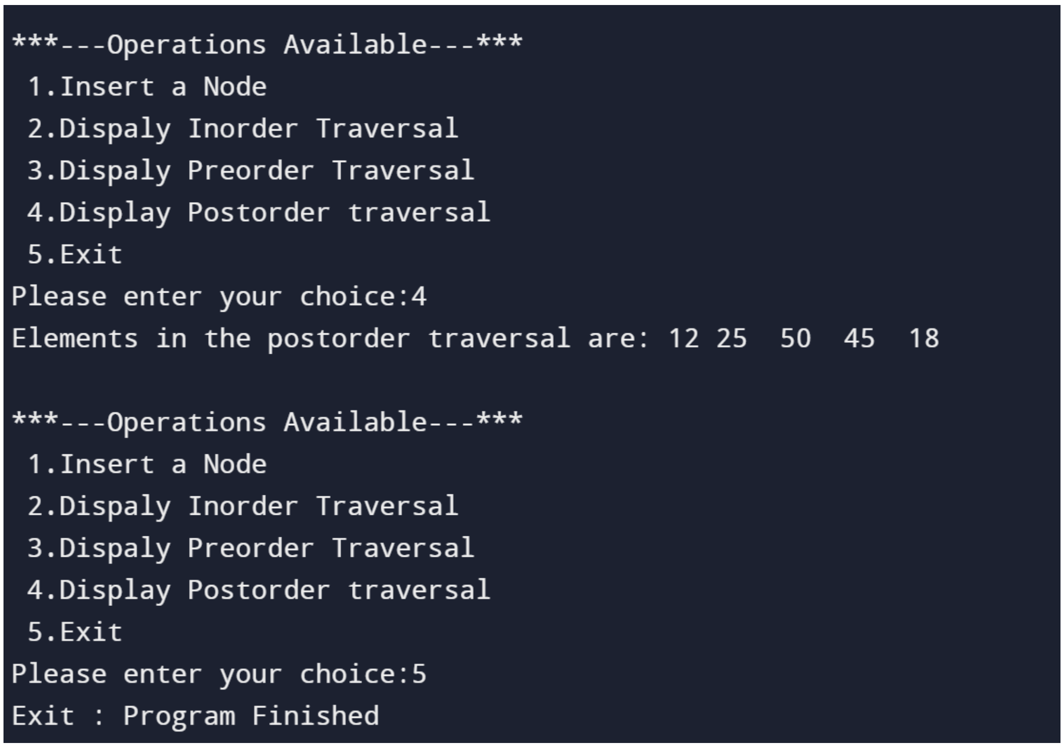
```
---Welcome To Implementation Of Binary Tree Traversals---

***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:1
Enter the data to be inserted:18
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:1
Enter the data to be inserted:45
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
```



```
Please enter your choice:1
Enter the data to be inserted:12
***---Operations Available---***
 1.Insert a Node2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:1
Enter the data to be inserted:25
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:1
Enter the data to be inserted:50
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
```



```
Please enter your choice:22
Please enter a valid option 1,2,3,4,5
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:2
Elements in the inorder traversal are: 12   18  25  45  50

***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:3
Elements in the preorder traversal are: 18  12  45  25  50
```



```
***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:4
Elements in the postorder traversal are: 12 25  50  45  18

***---Operations Available---***
 1.Insert a Node
 2.Dispaly Inorder Traversal
 3.Dispaly Preorder Traversal
 4.Display Postorder traversal
 5.Exit
Please enter your choice:5
Exit : Program Finished
```