**Program 6:** Design, develop and execute a program in C to create a Binary search tree and perform tree traversals.

```
#include<stdio.h>
#include <stdlib.h>
struct tnode
{
        int data;
        struct tnode *right,*left;
};
typedef struct tnode TNODE;

TNODE *CreateBST(TNODE *, int);
void Preorder(TNODE *);
void Postorder(TNODE *root);
void Inorder(TNODE *root);
int main( )
{
        TNODE *root=NULL;                   /* Main Program */
        int opn,elem,n,i;
        do
        {

        printf("\n Binary Search Tree Operations \n\n");
        printf("\n 1-Creation of BST\n 2. Inorder 3. Preorder 4. Postorder\n");
        printf("\n  Enter Your option ? ");
        scanf("%d",&opn);
        switch(opn)
        {
                case 1: root=NULL;
                printf("\n\nBST for How Many Nodes ?");
                scanf("%d",&n);
                for(i=1;i<=n;i++)
                {
                  printf("\nRead the Data for Node %d ?",i);
                  scanf("%d",&elem);
                  root=CreateBST(root,elem);
                }
                 printf("\nBST with %d nodes is ready to Use!!\n",n); break;

                case 2: printf("\n BST Traversal in INORDER \n");
                        Inorder(root); break;
                 case 3: printf("\n BST Traversal in PREORDER \n");
                        Preorder(root); break;
                 case 4: printf("\n BST Traversal in PostORDER \n");
                        Postorder(root); break;
```

```
                default: printf("\n\nInvalid Option !!! Try Again !! \n\n");  break;
        }
        printf("\n\n\n\n  Press a Key to Continue . . . ");
        }while(opn != 5);
}
TNODE *CreateBST(TNODE *root, int elem)
{
        if(root == NULL)
        {
        root=(TNODE *)malloc(sizeof(TNODE));
        root->left= root->right = NULL;
        root->data=elem;
        return root;
        }
        else
        {
        if( elem < root->data )
        root->left=CreateBST(root->left,elem);
        else
        if( elem > root->data )
                root->right=CreateBST(root->right,elem);
        else
                printf(" Duplicate Element !! Not Allowed !!!");

        return(root);
        }
}
void Preorder(TNODE *root)
{
        if( root != NULL)
        {
        printf(" %d ",root->data);
        Preorder(root->left);
        Preorder(root->right);
        }
}

void Postorder(TNODE *root)
{
        if( root != NULL)
        {
         Postorder(root->left);
        Postorder(root->right);
        printf(" %d ",root->data);
        }
}
```

```
void Inorder(TNODE *root)
{
        if( root != NULL)
        {
         Inorder(root->left);
        printf(" %d ",root->data);
        Inorder(root->right);

        }
}
```