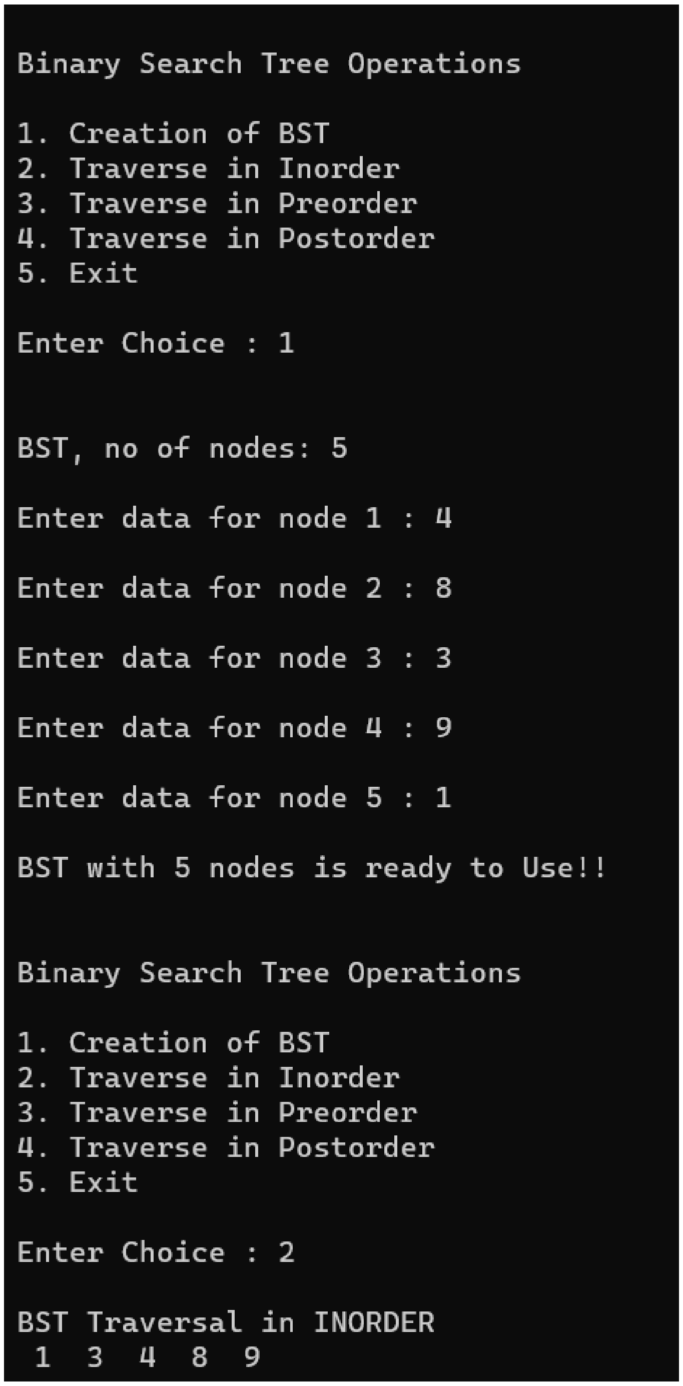
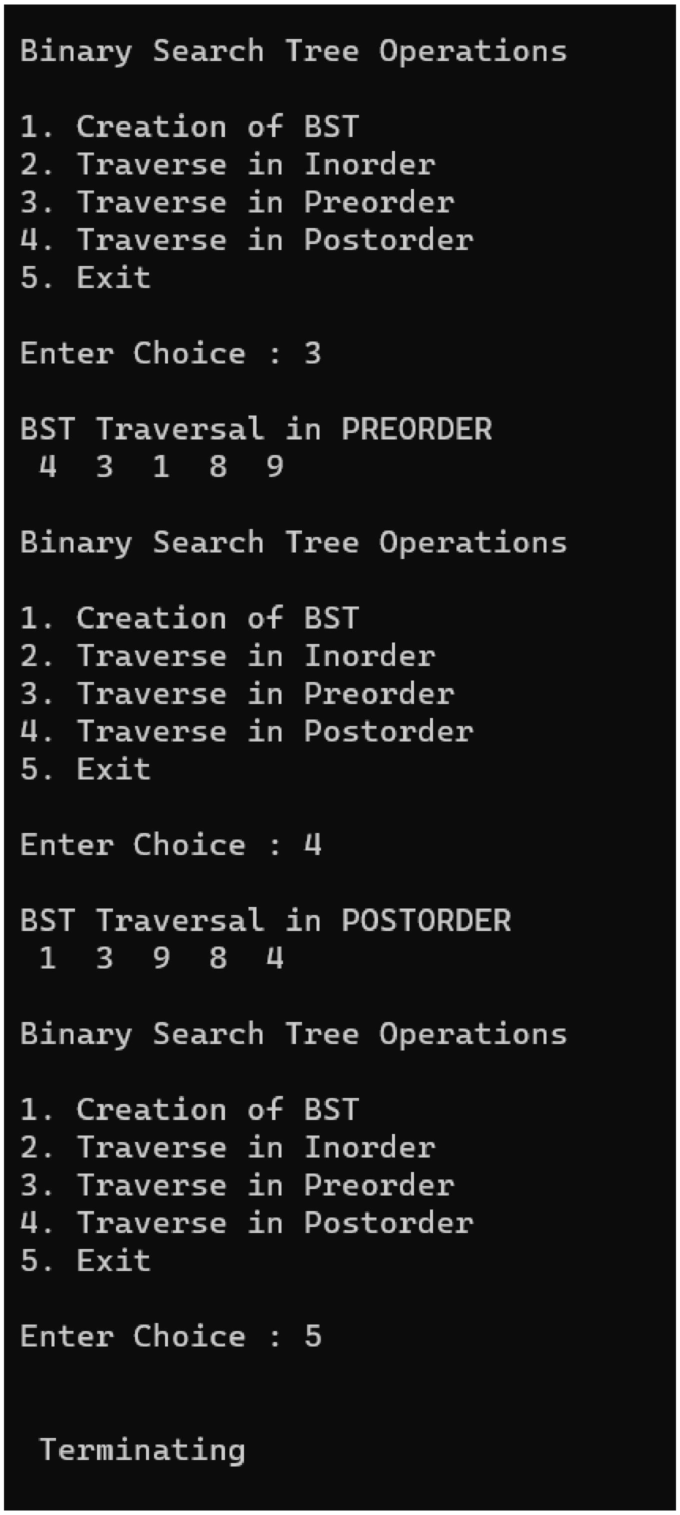
PROGRAM

```
#include <stdio.h>
#include <stdlib.h>
//Creating structure
struct tnode{
 int data;
 struct tnode *right;
 struct tnode *left;};
//Function to insert new node to the BST
struct tnode *CreateBST(struct tnode *root, int item){
 if(root == NULL){
//Allocating memory to the new node
 root = (struct tnode *)malloc(sizeof(struct tnode));
 root->left = root->right = NULL;
 root->data = item;
 return root; }
 else{
 if(item < root->data )
 root->left = CreateBST(root->left,item);

 else if(item > root->data )
 root->right = CreateBST(root->right,item);
 else    printf(" Duplicate Element !! Not Allowed !!!");
 return(root); } }
//Function to perform inorder traversal
void Inorder(struct tnode *root){
 if( root != NULL) {
 Inorder(root->left);
 printf(" %d ",root->data);
 Inorder(root->right); } }
//Function to perform preorder traversal
void Preorder(struct tnode *root){
 if( root != NULL){
 printf(" %d ",root->data);
 Preorder(root->left);
 Preorder(root->right); }}
//Function to perform postorder traversal
void Postorder(struct tnode *root){
 if( root != NULL) {
 Postorder(root->left);
 Postorder(root->right);
 printf(" %d ",root->data); }}
void main(){
 struct tnode *root = NULL;
 int choice, item, n, i;
do {
 printf("\n\nBinary Search Tree Operations\n");
 printf("\n1. Creation of BST");
 printf("\n2. Traverse in Inorder");
 printf("\n3. Traverse in Preorder");
 printf("\n4. Traverse in Postorder");
 printf("\n5. Exit\n");
 printf("\nEnter Choice : ");
 scanf("%d",&choice);
 switch(choice){
 case 1:
 root = NULL;
 printf("\n\nBST, no of nodes: ");
 scanf("%d",&n);
 for(i = 1; i <= n; i++) {
 printf("\nEnter data for node %d : ", i);
 scanf("%d",&item);
 root = CreateBST(root,item); }
 printf("\nBST with %d nodes is ready to Use!!\n", n);    break;
 case 2:
 printf("\nBST Traversal in INORDER \n");
 Inorder(root);    break;
 case 3:
 printf("\nBST Traversal in PREORDER \n");
 Preorder(root);   break;
 case 4:
 printf("\nBST Traversal in POSTORDER \n");
 Postorder(root);    break;
 case 5:
 printf("\n\n Terminating \n\n");   break;
 default:
 printf("\n\nInvalid Option !!! Try Again !! \n\n");
 break; } }   while(choice != 5);}
```

OUTPUT



```
Binary Search Tree Operations

1. Creation of BST
2. Traverse in Inorder
3. Traverse in Preorder
4. Traverse in Postorder
5. Exit

Enter Choice : 1


BST, no of nodes: 5

Enter data for node 1 : 4

Enter data for node 2 : 8

Enter data for node 3 : 3

Enter data for node 4 : 9

Enter data for node 5 : 1

BST with 5 nodes is ready to Use!!


Binary Search Tree Operations

1. Creation of BST
2. Traverse in Inorder
3. Traverse in Preorder
4. Traverse in Postorder
5. Exit

Enter Choice : 2

BST Traversal in INORDER
 1  3  4  8  9
```



```
Binary Search Tree Operations

1. Creation of BST
2. Traverse in Inorder
3. Traverse in Preorder
4. Traverse in Postorder
5. Exit

Enter Choice : 3

BST Traversal in PREORDER
 4  3  1  8  9

Binary Search Tree Operations

1. Creation of BST
2. Traverse in Inorder
3. Traverse in Preorder
4. Traverse in Postorder
5. Exit

Enter Choice : 4

BST Traversal in POSTORDER
 1  3  9  8  4

Binary Search Tree Operations

1. Creation of BST
2. Traverse in Inorder
3. Traverse in Preorder
4. Traverse in Postorder
5. Exit

Enter Choice : 5


 Terminating
```