```
//Created By Ritwik Chandra Pandey on 30/3/21
//183215
//BST Tree-Traversal&Search


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

struct node {
    int key;
    struct node *left, *right;
};

// A utility function to create a new BST node
struct node* newNode(int item)
{
    struct node* temp
        = (struct node*)malloc(sizeof(struct node));
    temp->key = item;
    temp->left = temp->right = NULL;
    return temp;
}

void Preorder(struct node* t) {
if(t!=NULL) {
printf("%d ",t->key);
Preorder(t->left);
Preorder(t->right);}
}

void Postorder(struct node* t) {
if( t!=NULL){
Postorder(t->left);
Postorder(t->right);
printf("%d ",t->key);
```

```
}
}


// A utility function to do inorder traversal of BST
void inorder(struct node* root)
{
    if (root != NULL) {
        inorder(root->left);
        printf("%d ", root->key);
        inorder(root->right);
    }
}

/* A utility function to insert
   a new node with given key in
 * BST */
struct node* insert(struct node *node, int key)
{
    /* If the tree is empty, return a new node */
    if (node == NULL)
        node=newNode(key);

    /* Otherwise, recur down the tree */
    if (key < node->key)
        node->left = insert(node->left, key);
    else if (key > node->key)
        node->right = insert(node->right, key);

    /* return the (unchanged) node pointer */
    return node;
}
struct node* find(int x,struct node* root){
    if(root==NULL){
        return root;
    }else if(x<root->key){
        return find(x,root->left);
```

```
    }else if(x>root->key){
        return find(x,root->right);
    }
    else return root;
}
// Driver Code
int main()
{int x=0;


    struct node* root = NULL;
    struct node* temp;

    printf("Enter elements to be entered for the binary search tree: (-1 stops input) ");
    scanf("%d",&x);
    while(x!=-1){
        root=insert(root,x);
        scanf("%d",&x);

    }


    printf("Inorder Traversal of binary search tree : ");
    inorder(root);

    printf("\n");
    printf("Preorder Traversal of binary search tree : ");
    Preorder(root) ;

    printf("\n");

    printf("Postorder Traversal of binary search tree : ");
    Postorder(root);

    printf("\n");
    printf("Enter the element that you want to search for in the BST : ");
    scanf("%d",&x);
```

```
    temp = find(x,root);
    if(temp==NULL){
        printf("The given value %d is not present in the BST.\n",x);
    }else{
   printf("The given value %d is present in the BST and its node has been secured.\n",temp->key);
    }//The node has been secured in temp.


    return 0;
}
```