```
//Created By Ritwik Pandey on 27th Sep 2021
//Creation, Insertion and In-order traversal



#include<stdio.h>
#include<stdlib.h>
struct node {
	int data;
	struct node *left, *right;
};

typedef struct node *BSTNODE;

BSTNODE newNodeInBST(int item) {
	BSTNODE temp =  (BSTNODE)malloc(sizeof(struct node));
	temp->data = item;
	temp->left = temp->right = NULL;
	return temp;
}

void inorderInBST(BSTNODE root) {
	if(root==NULL) return;
	inorderInBST(root->left);
	printf("%d ",root->data);
	inorderInBST(root->right);

}

BSTNODE insertNodeInBST(BSTNODE node, int ele) {
	if(node==NULL){
		printf("Successfully inserted.\n");
		return newNodeInBST(ele);
	}
	if(ele < node->data){
		node->left = insertNodeInBST(node->left,ele);
	}else if(ele> node->data){
		node->right = insertNodeInBST(node->right,ele);
	}
	else
```

```
		printf("Element already exists in BST.\n");
		return node;
	}

	void main() {
		int x, op;
		BSTNODE root = NULL;
		while(1)
		{
			printf("1.Insert 2.Inorder Traversal 3.Exit\n");
			printf("Enter your option : ");
			scanf("%d", &op);
			switch(op) {
				case 1:printf("Enter an element to be inserted : ");
						scanf("%d", &x);
						root = insertNodeInBST(root,x);
						break;
				case 2:
						if(root == NULL) {
							printf("Binary Search Tree is empty.\n");
						}
						else {
							printf("Elements of the tree (in-order traversal): ");
							inorderInBST(root);
							printf("\n");
							}
						break;
				case 3:
						exit(0);
			}
		}
	}
```