S.No: 22	Exp. Name: *Program to insert into BST and traversal using In-order, Pre-order and Post-order*	Date:2023-06-11

Aim:

Write a program to create a binary search tree of integers and perform the following operations using linked list.

1. Insert a node
2. In-order traversal
3. Pre-order traversal
4. Post-order traversal

Source Code:

BinarySearchTree.c

```c
#include<stdio.h>
#include<stdlib.h>
#include "InsertAndTraversals.c"

void main() {
    int x, op;
    BSTNODE root = NULL;
    while(1) {
        printf("1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.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 BST (in-order traversal): ");
                        inorderInBST(root);
                        printf("\n");
                    }
                    break;
            case 3:
                    if(root == NULL) {
                        printf("Binary Search Tree is empty.\n");
                    }
                    else {
                        printf("Elements of the BST (pre-order traversal): ");
                        preorderInBST(root);
                        printf("\n");
                    }
                    break;
            case 4:
                    if(root == NULL) {
                        printf("Binary Search Tree is empty.\n");
```

```
                }
                else {
                    printf("Elements of the BST (post-order traversal): ");
                    postorderInBST(root);
                    printf("\n");
                }
                break;
        case 5:
                exit(0);
        }
    }
}
```

InsertAndTraversals.c

```c
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)
    {
        inorderInBST(root->left);
        printf("%d ",root->data);
        inorderInBST(root->right);
    }

}

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

}

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

```
        }

    }
    BSTNODE insertNodeInBST(BSTNODE node, int ele) {
        if(node==NULL)
        {
            printf("Successfully inserted.\n");
            return newNodeInBST(ele);
        }
        else 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;

    }
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 54
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 28
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 62
Successfully inserted. 2
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 2
Enter your option : 2
Elements of the BST (in-order traversal): 28 54 62 3
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 3
Enter your option : 3
Elements of the BST (pre-order traversal): 54 28 62 4
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 4
Enter your option : 4
Elements of the BST (post-order traversal): 28 62 54 5
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 5
Enter your option : 5

Test Case - 2
User Output
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 100
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 20
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 200
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 10
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 30
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 150
Successfully inserted. 1
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 1
Enter your option : 1
Enter an element to be inserted : 300
Successfully inserted. 2
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 2
Enter your option : 2
Elements of the BST (in-order traversal): 10 20 30 100 150 200 300 3
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 3
Enter your option : 3
Elements of the BST (pre-order traversal): 100 20 10 30 200 150 300 4
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 4
Enter your option : 4
Elements of the BST (post-order traversal): 10 30 20 150 300 200 100 5
1.Insert 2.Inorder Traversal 3.Preorder Traversal 4.Postorder Traversal 5.Exit 5
Enter your option : 5