LAB PROGRAM 8

Write a program a) To construct a binary Search tree. b) To traverse the tree using all the methods i.e., in-order, preorder and post order c) To display the elements in the tree.

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

struct node {
    int data;
    struct node *left;
    struct node *right;
};

struct node* createNode(int x) {
    struct node *newnode = (struct node *)malloc(sizeof(struct node));
    newnode->data = x;
    newnode->left = NULL;
    newnode->right = NULL;
    return newnode;
}

struct node* insert(struct node *root, int x) {
    if (root == NULL)
        return createNode(x);

    if (x < root->data)
        root->left = insert(root->left, x);
    else
        root->right = insert(root->right, x);
```

```
    return root;
}

void inorder(struct node *root) {
    if (root != NULL) {
        inorder(root->left);
        printf("%d ", root->data);
        inorder(root->right);
    }
}

void preorder(struct node *root) {
    if (root != NULL) {
        printf("%d ", root->data);
        preorder(root->left);
        preorder(root->right);
    }
}

void postorder(struct node *root) {
    if (root != NULL) {
        postorder(root->left);
        postorder(root->right);
        printf("%d ", root->data);
    }
}

int main() {
```

```
struct node *root = NULL;
int n, x, choice;

printf("Enter number of nodes: ");
scanf("%d", &n);

for (int i = 0; i < n; i++) {
    scanf("%d", &x);
    root = insert(root, x);
}

while (1) {
    printf("\n1.Inorder\n2.Preorder\n3.Postorder\n4.Exit\n");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            inorder(root);
            printf("\n");
            break;
        case 2:
            preorder(root);
            printf("\n");
            break;
        case 3:
            postorder(root);
            printf("\n");
            break;
        case 4:
```

```
            exit(0);
        default:
            printf("Invalid choice\n");
    }
  }
}
```

OUTPUT:

```
Enter number of nodes:3

11

12

13


1.Inorder
2.Preorder
3.Postorder
4.Exit
1

11 12 13
```

```
1.Inorder
2.Preorder
3.Postorder
4.Exit
2

11 12 13

1.Inorder
2.Preorder
3.Postorder
4.Exit
3

13 12 11
```