Program 9

Write a program

- 8a) To construct a binary Search tree.
- 8b) To traverse the tree using all the methods i.e., in-order, preorder and post order, display all traversal order

Observation:

```
NEER - 10

8. Write a program

1) To construct a binary search time.

1) To traverse the tree using all the methods

i.e., in-order, preorder, and past order display all

traversal order.

90) Write a program to traverse a graph using

BES method.

96) Write a program to traverse a graph using

DES method.

8 struct Node f

Int data;

struct Node repate Node (int data);

struct Node recate Node (int data);

struct Node recate Node (int data);

reworked - I data = data;

newNode - I data = data;

newNode - I left: NULL;

newNode - I right: NULL;

return resolved;

stewat Node enert (struct Node root, int data);

return create Node (data);

if (data & root-ydata);

root-> left = insert (root-> left, data);

2 root-> left = insert (root-> left, data);
```

```
else st (data > mot -> data) {

not -> right = Prisert (mot -> right, data);

return mot;

}

void Priorder (struct Node* mot) {

lef (mot!=Nou) {

Priorder (mot -> left),

Printf ("Id", mot -> data);

Printf ("Not -> left);

Printf ("Not -> left);

Printf ("Not -> left);

Priorder (mot -> right);

If (mot!=Nou) {

priorder (mot -> right);

Priorder (mot -> right);

Priorder (mot -> right);

postorder (mot -> left);

postorder (mo
```

```
Desplay For Order Froversal

Enter your choice:1

Enter your choice:1

Enter your choice:1

Enter the value to insert: 35

Enter your choice:1

Enter the value to insert: 35

Enter your choice:1

Enter your choice:3

Enter your choice:3

Enter your choice:3

Pre order Troversal:15 at as 30 35 40

Enter your choice:4

Pre order Troversal:68 15 20 28 28 40

Enter your Choice:4

Pre order Troversal:15 25 25 28 38 40

Enter your choice:4

Pre order Troversal:15 25 25 28 38 40
```

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct Node {
   int data;
    struct Node* left;
    struct Node* right;
};
struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->left = NULL;
    newNode->right = NULL;
    return newNode;
struct Node* insert(struct Node* root, int data) {
    if (root == NULL) {
        return createNode(data);
    if (data < root->data) {
        root->left = insert(root->left, data);
    } else if (data > root->data) {
        root->right = insert(root->right, data);
    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 choice, value;
    printf("Binary Search Tree Operations:\n");
    printf("1. Insert a node\n");
    printf("2. Display In-order Traversal\n");
    printf("3. Display Pre-order Traversal\n");
    printf("4. Display Post-order Traversal\n");
    printf("5. Exit\n");
    while (1) {
        printf("\nEnter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("Enter the value to insert: ");
                scanf("%d", &value);
                root = insert(root, value);
                break;
            case 2:
                printf("In-order Traversal: ");
                inorder(root);
                printf("\n");
                break;
            case 3:
                printf("Pre-order Traversal: ");
                preorder(root);
                printf("\n");
                break;
            case 4:
                printf("Post-order Traversal: ");
                postorder(root);
                printf("\n");
                break;
            case 5:
                printf("Exiting...\n");
                exit(0);
            default:
```

```
printf("Invalid choice! Please try again.\n");
}
}
return 0;
}
```

Output:

```
Binary Search Tree Operations:
1. Insert a node
2. Display In-order Traversal
3. Display Pre-order Traversal
4. Display Post-order Traversal
5. Exit
Enter your choice: 1
Enter the value to insert: 20
Enter your choice: 1
Enter the value to insert: 30
Enter your choice: 1
Enter the value to insert: 35
Enter your choice: 1
Enter the value to insert: 40
Enter your choice: 1
Enter the value to insert: 25
Enter your choice: 1
Enter the value to insert: 15
Enter your choice: 2
In-order Traversal: 15 20 25 30 35 40
Enter your choice: 3
Pre-order Traversal: 20 15 30 25 35 40
Enter your choice: 4
Post-order Traversal: 15 25 40 35 30 20
```