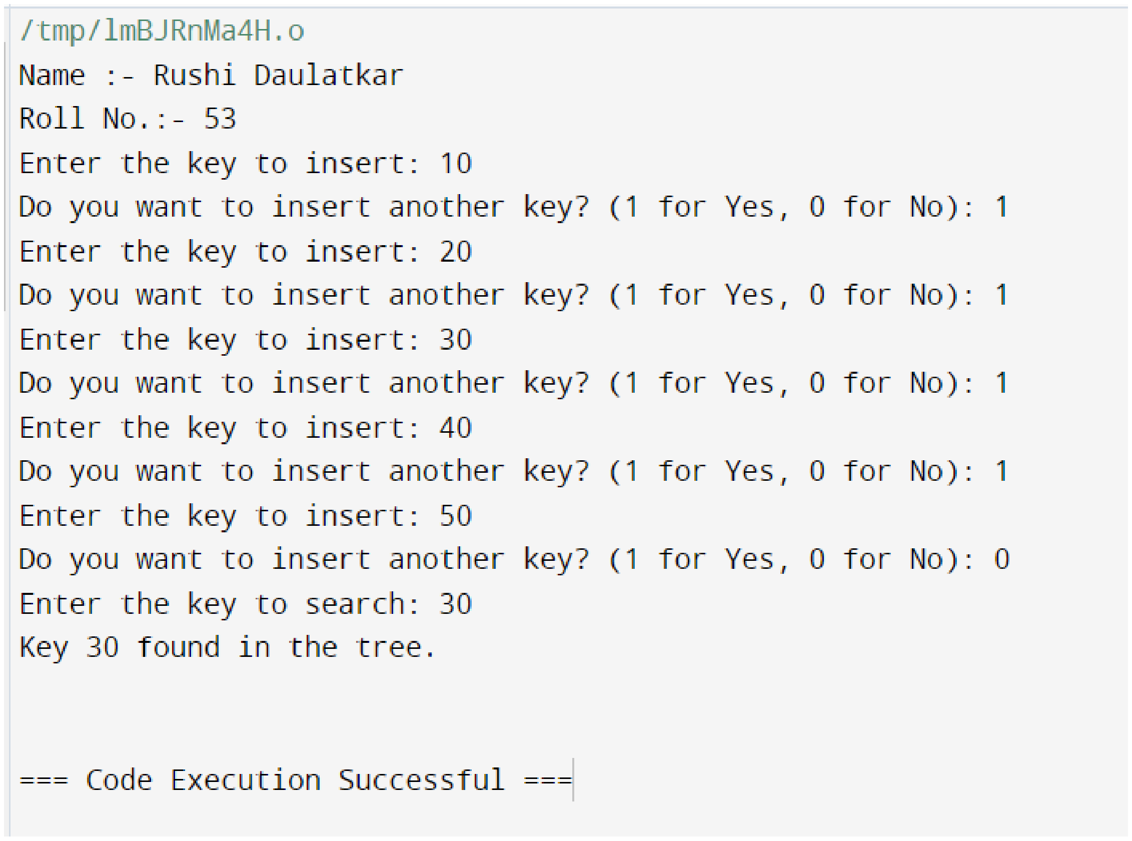
EXPERIMENT NO. 12.(A)

//The Code Is As Follows :-

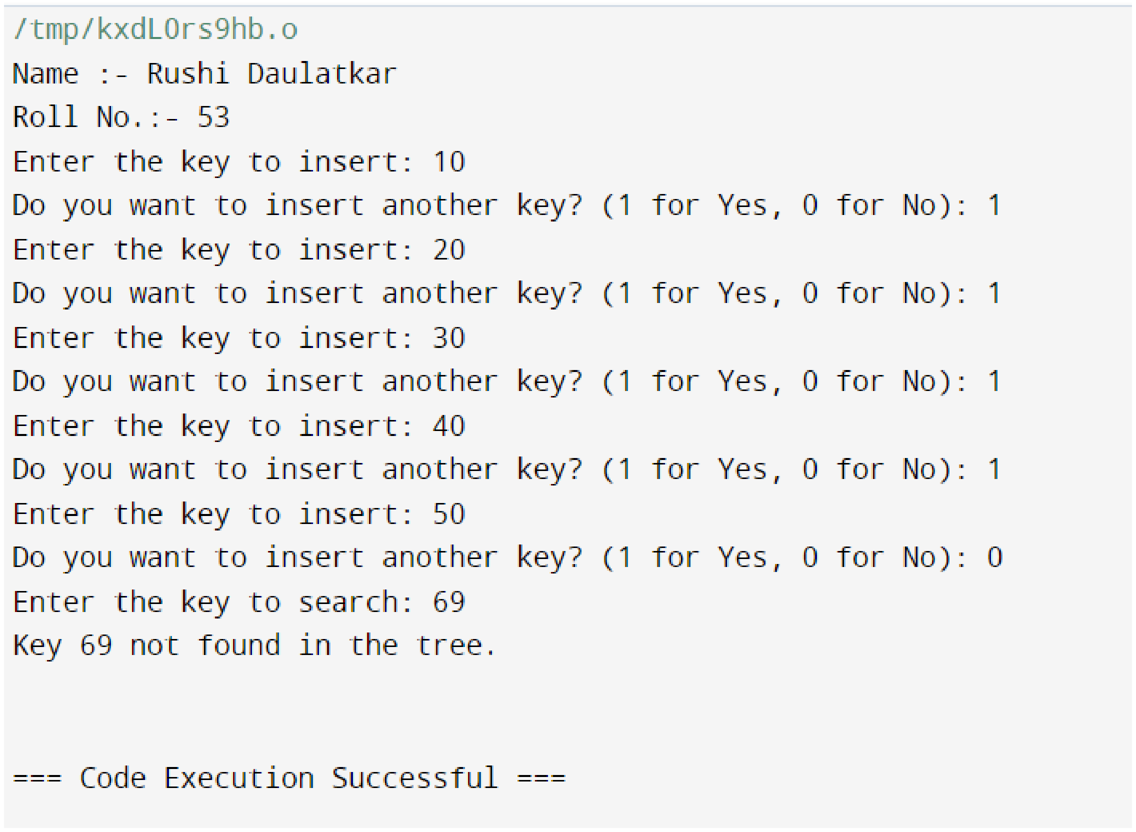
```
#include <stdio.h>
#include <stdlib.h>

// Define the structure for a tree node
struct TreeNode {
    int key;                    // Key value of the node
    struct TreeNode *left;      // Pointer to the left child node
    struct TreeNode *right;     // Pointer to the right child node
};

// Function to create a new node with given key
struct TreeNode* newNode(int key) {
    // Allocate memory for a new node
    struct TreeNode* node = (struct TreeNode*) malloc(sizeof(struct TreeNode));
    // Assign the key value
    node->key = key;
    // Initialize left and right child pointers as NULL
    node->left = NULL;
    node->right = NULL;
    // Return the newly created node
    return node;
```

```c
}

// Function to insert a new key in BST
struct TreeNode* insert(struct TreeNode* node, int key) {
    // If the tree is empty, return a new node
    if (node == NULL) return newNode(key);

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

    // Return the (unchanged) node pointer
    return node;
}

// Function to search a given key in BST
struct TreeNode* search(struct TreeNode* root, int key) {
    // Base cases: root is null or key is present at root
    if (root == NULL || root->key == key)
       return root;
    // Key is greater than root's key
    if (root->key < key)
       // Search in the right subtree recursively
       return search(root->right, key);

    // Key is smaller than root's key
    // Search in the left subtree recursively
    return search(root->left, key);
}

// Main function to test above functions
int main() {
printf("Name :- Rushi Daulatkar \n");
printf("Roll No.:- 53 \n");

    struct TreeNode* root = NULL;
    int key, choice;

    // Loop to insert keys into the BST
    do {
        // Prompt user to enter the key to insert
        printf("Enter the key to insert: ");
        // Read the key from user input
        scanf("%d", &key);
        // Insert the key into the BST
        root = insert(root, key);


        // Ask user if they want to insert another key
        printf("Do you want to insert another key? (1 for Yes, 0 for No): ");
        // Read the choice from user input
        scanf("%d", &choice);
    } while(choice != 0);


    // Variable to store the key to search
    int searchKey;
    // Prompt user to enter the key to search
    printf("Enter the key to search: ");
    // Read the key from user input
    scanf("%d", &searchKey);
    // Search for the key in the BST
    struct TreeNode* found = search(root, searchKey);
    // Check if the key is found and print appropriate message
    if(found)
        printf("Key %d found in the tree.\n", searchKey);
    else
        printf("Key %d not found in the tree.\n", searchKey);


    return 0;
}
```

OUTPUT:-



```
/tmp/lmBJRnMa4H.o
Name :- Rushi Daulatkar
Roll No.:- 53
Enter the key to insert: 10
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 20
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 30
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 40
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 50
Do you want to insert another key? (1 for Yes, 0 for No): 0
Enter the key to search: 30
Key 30 found in the tree.


=== Code Execution Successful ===
```



```
/tmp/kxdL0rs9hb.o
Name :- Rushi Daulatkar
Roll No.:- 53
Enter the key to insert: 10
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 20
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 30
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 40
Do you want to insert another key? (1 for Yes, 0 for No): 1
Enter the key to insert: 50
Do you want to insert another key? (1 for Yes, 0 for No): 0
Enter the key to search: 69
Key 69 not found in the tree.


=== Code Execution Successful ===
```