<u>Experiment 4:</u> Write a C program that uses functions to per form the following: a) Create a binary search tree of characters.

experiment4a.c

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct Node {
    char data;
    struct Node* left;
    struct Node* right;
} Node;
Node* createNode(char data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    if (newNode == NULL) {
        printf("Memory error\n");
        return NULL;
    }
    newNode->data = data;
    newNode->left = NULL;
    newNode->right = NULL;
    return newNode;
}
Node* insert(Node* root, char 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;
}
Node* search(Node* root, char data) {
    if (root == NULL || root->data == data) {
        return root;
    }

    if (data < root->data) {
        return search(root->left, data);
    } else {
        return search(root->right, data);
    }
}
void inorderTraversal(Node* root) {
    if (root != NULL) {
        inorderTraversal(root->left);
        printf("%c ", root->data);
        inorderTraversal(root->right);
    }
}
```

```c
int main() {
    Node* root = NULL;
    root = insert(root, 'F');
    insert(root, 'D');
    insert(root, 'J');
    insert(root, 'B');
    insert(root, 'E');
    insert(root, 'G');
    insert(root, 'K');
    printf("Inorder traversal of the binary search tree: ");
    inorderTraversal(root);
    printf("\n");
    char searchChar = 'E';
    Node* result = search(root, searchChar);
    if (result != NULL) {
        printf("Character '%c' found in the tree.\n", searchChar);
    } else {
        printf("Character '%c' not found in the tree.\n", searchChar);
    }
    free(root);
    return 0;
}
```

Output:

```
D:\SubhamDasDSUC\experim

Inorder traversal of the binary search tree: B D E F G J K
Character 'E' found in the tree.


--------------------------------
Process exited after 0.02791 seconds with return value 0
Press any key to continue . . .
```

b) Traverse the above Binary search tree recursively in Post order.

```c
#include <stdio.h>
#include <stdlib.h>
typedef struct TreeNode {
    char data;
    struct TreeNode *left;
    struct TreeNode *right;
} TreeNode;
TreeNode *createNode(char data) {
    TreeNode *newNode = (TreeNode *)malloc(sizeof(TreeNode));
    newNode->data = data;
    newNode->left = newNode->right = NULL;
    return newNode;
}
TreeNode *insert(TreeNode *root, char 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;
}
```

```c
void postOrderTraversal(TreeNode *root) {
    if (root == NULL)
        return;
    postOrderTraversal(root->left);
    postOrderTraversal(root->right);
    printf("%c ", root->data);
}
int main() {
    int i;
    TreeNode *root = NULL;
    char charArray[] = {'F', 'D', 'J', 'B', 'E', 'G', 'K', 'A', 'C', 'I', 'H'};
    int numChars = sizeof(charArray) / sizeof(charArray[0]);
    for (i = 0; i < numChars; i++) {
        root = insert(root, charArray[i]);
    }
    printf("Post-order traversal:\n");
    postOrderTraversal(root);
    printf("\n");
    return 0;
}
```

Output:

```
D:\SubhamDasDSUC\experim
Post-order traversal:
A C B E D H I G K J F

--------------------------------
Process exited after 0.02224 seconds with return value 0
Press any key to continue . . .
```