VISHNU B GURAV

642403005

ASSIGNMENT 5

header.h

```
typedef struct Node{
        char month[10];
        struct Node *lchild;
        struct Node *rchild;
        struct Node *parent;
}Node;

typedef struct BST {
        Node *t;
} BST;

void initBST(BST* tree);
void insertnode(BST *tree, char *month);
void removenode(BST *tree, char *month);
void traverse(BST *tree);
void destroyTree(Node *n);
void destroyBST(BST *tree);
```

logic.c

```
#include<stdio.h>
#include<stdlib.h>
#include <string.h>
#include "header.h"

void initBST(BST *tree){
    tree->t = NULL;
}

Node* createNode(char *month){
    Node *nn = (Node*)malloc(sizeof(Node));
    strcpy(nn->month, month);
    nn->lchild = nn->rchild = nn->parent = NULL;
    return nn;
}

void insertnode(BST *tree, char *month){
    Node *nn = createNode(month);
    if (tree->t== NULL){
        tree->t = nn;
        return;
    }
    Node *current = tree->t;
    Node *parent = NULL;
    while (current != NULL){
        parent = current;
        if (strcmp(month, current->month) < 0) {
            current = current->lchild;
```

```
        }else{
            current = current->rchild;
        }
    }
    nn->parent = parent;
    if (strcmp(month, parent->month) < 0){
        parent->lchild= nn;
    }else{
        parent->rchild = nn;
    }
    return ;
}

Node* findMin(Node *node){
    while (node->lchild != NULL){
        node = node->lchild;
    }
    return node;
}

void removenode(BST *tree, char *month){
    Node *current = tree->t;
    Node *parent = NULL;

    while (current != NULL && strcmp(current->month, month) != 0){
        parent = current;
        if (strcmp(month, current->month) < 0) {
            current = current->lchild;
        } else {
            current = current->rchild;
```

```
    }
}

if (current == NULL) {
    printf("Node with month %s not found.\n", month);
    return;
}

Node *child;
if (current->lchild == NULL || current->rchild == NULL){
    child = current->lchild ? current->lchild : current->rchild;



    if (parent == NULL){
        tree->t = child;
    }
    else if (parent->lchild == current){
        parent->lchild = child;
    } else {
        parent->rchild= child;
    }

    if (child != NULL){
        child->parent = parent;
    }
    free(current);
}

else{
    Node *successor = findMin(current->rchild);
```

```
        strcpy(current->month, successor->month);
        removenode(tree, successor->month);
    }

    printf("Node with month %s removed.\n", month);
    return ;
}

void traverse(BST *tree){
    if (tree->t == NULL) return;

    Node *current = tree->t;
    Node *stack[100];
    int top = -1;

    while (current != NULL || top != -1){

        while (current != NULL) {
            stack[++top] = current;
            current = current->lchild;
        }

        current = stack[top--];
        printf("%s ", current->month);

        current = current->rchild;
    }
    return ;
}
```

```
void destroyTree(Node *node){
    if (node == NULL) {
        return;
    }
    destroyTree(node->lchild);
    destroyTree(node->rchild);
    free(node);
}

void destroyBST(BST *tree){
    destroyTree(tree->t);
    tree->t = NULL;
    printf("Tree destroyed.\n");
}
```

main.c

```
#include<stdio.h>
#include<stdlib.h>
#include <string.h>
#include "header.h"

int main() {
    BST tree;
    initBST(&tree);

    int choice;
    char month[10];
```

```
do {
    printf("\nMenu:\n");
    printf("1. Insert Node\n");
    printf("2. Remove Node\n");
    printf("3. Traverse (In-order)\n");
    printf("4. Destroy Tree\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            printf("Enter month to insert: ");
            scanf("%s", month);
            insertnode(&tree, month);
            printf("Node with month %s inserted.\n", month);
            break;

        case 2:
            printf("Enter month to remove: ");
            scanf("%s", month);
            removenode(&tree, month);
            break;

        case 3:
            printf("In-order traversal of the tree: ");
            traverse(&tree);
            break;
```

```
            case 4:
                destroyBST(&tree);
                break;

            default:
                printf("Invalid choice! Please try again.\n");
        }
    } while (choice != 5);

    return 0;
}
```

Output

```
DEBUG CONSOLE   PROBLEMS   OUTPUT   TERMINAL   PORTS

Enter month to insert: Node with month may inserted.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: june
Enter month to insert: Node with month june inserted.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree: april january june march may novemeber
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 2
Enter month to remove: june
Node with month june removed.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree: april january march may novemeber
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
```


```
DEBUG CONSOLE   PROBLEMS   OUTPUT   TERMINAL   PORTS

Enter month to insert: Node with month may inserted.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: june
Enter month to insert: Node with month june inserted.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree: april january june march may novemeber
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 2
Enter month to remove: june
Node with month june removed.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree: april january march may novemeber
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
```

```
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree: april january march may novemeber
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 4
Tree destroyed.

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 3
In-order traversal of the tree:
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
```