Nisarg Wath
642403014

Assignment 5

HEADER.H

```c
typedef struct TreeNode {
    char month[10];
    struct TreeNode *left;
    struct TreeNode *right;
    struct TreeNode *parent;
} TreeNode;

typedef struct BST {
    TreeNode *root;
} BST;


void initBST(BST* tr);

void insertNode(BST *tree, char *month);
void removeNode(BST *tree, char *month);
void traverse(BST *tree);

void destroyTree(TreeNode *node);
void destroyBST(BST *tree);
```

LOGIC.C

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

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

TreeNode* createNode(char *month) {
    TreeNode *newNode = (TreeNode*)malloc(sizeof(TreeNode));
    strcpy(newNode->month, month);
    newNode->left = newNode->right = newNode->parent = NULL;
    return newNode;
}


void insertNode(BST *tree, char *month) {
```

```
    TreeNode *newNode = createNode(month);
    if (tree->root == NULL) {
        tree->root = newNode;
        return;
    }

    TreeNode *current = tree->root;
    TreeNode *parent = NULL;

    while (current != NULL) {
        parent = current;
        if (strcmp(month, current->month) < 0) {
            current = current->left;
        } else {
            current = current->right;
        }
    }

    newNode->parent = parent;
    if (strcmp(month, parent->month) < 0) {
        parent->left = newNode;
    } else {
        parent->right = newNode;
    }
}

TreeNode* findMin(TreeNode *node) {
    while (node->left != NULL) {
        node = node->left;
    }
    return node;
}


void removeNode(BST *tree, char *month) {
    TreeNode *current = tree->root;
    TreeNode *parent = NULL;


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

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

```
        return;
    }


    TreeNode *child;
    if (current->left == NULL || current->right == NULL) {
        child = current->left ? current->left : current->right;


        if (parent == NULL) {
            tree->root = child;
        }

        else if (parent->left == current) {
            parent->left = child;
        } else {
            parent->right = child;
        }

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

    else {
        TreeNode *successor = findMin(current->right);
        strcpy(current->month, successor->month);
        removeNode(tree, successor->month);
    }

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


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

    TreeNode *current = tree->root;
    TreeNode *stack[100];
    int top = -1;

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

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


        current = stack[top--];
```

```
        printf("%s ", current->month);


        current = current->right;
    }
}


void destroyTree(TreeNode *node) {
    if (node == NULL) {
        return;
    }
    destroyTree(node->left);
    destroyTree(node->right);
    free(node);
}
void destroyBST(BST *tree) {
    destroyTree(tree->root);
    tree->root = NULL;
    printf("Tree destroyed.\n");
}
```

}Main.c

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

```
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 1
Enter month to insert:  December, January, April, March, July, August, October, February, November, May, June
Node with month December, inserted.

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

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

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

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

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

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

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

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

Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
Enter your choice: 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: 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, August, December, February, January, July, June March, May, November, October,
Menu:
1. Insert Node
2. Remove Node
3. Traverse (In-order)
4. Destroy Tree
5. Exit
```