#include <stdio.h>

#include <stdlib.h>

typedef struct Node {

int data;

struct Node\* left;

struct Node\* right;

} Node;

Node\* createNode(int data) {

Node\* newNode = (Node\*)malloc(sizeof(Node));

newNode->data = data;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

Node\* insert(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;

}

Node\* findMin(Node\* node) {

Node\* current = node;

while (current && current->left != NULL) {

current = current->left;

}

return current;

}

Node\* deleteNode(Node\* root, int data) {

if (root == NULL) {

return root;

}

if (data < root->data) {

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

} else if (data > root->data) {

root->right = deleteNode(root->right, data);

} else {

if (root->left == NULL) {

Node\* temp = root->right;

free(root);

return temp;

} else if (root->right == NULL) {

Node\* temp = root->left;

free(root);

return temp;

}

Node\* temp = findMin(root->right);

root->data = temp->data;

root->right = deleteNode(root->right, temp->data);

}

return root;

}

void inOrder(Node\* root) {

if (root != NULL) {

inOrder(root->left);

printf("%d ", root->data);

inOrder(root->right);

}

}

void preOrder(Node\* root) {

if (root != NULL) {

printf("%d ", root->data);

preOrder(root->left);

preOrder(root->right);

}

}

void postOrder(Node\* root) {

if (root != NULL) {

postOrder(root->left);

postOrder(root->right);

printf("%d ", root->data);

}

}

int menu() {

int choice;

printf("\nBinary Search Tree Menu\n");

printf("1. Insert Element\n");

printf("2. Delete Element\n");

printf("3. In-order Traversal\n");

printf("4. Pre-order Traversal\n");

printf("5. Post-order Traversal\n");

printf("6. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

return choice;

}

int main() {

Node\* root = NULL;

int choice, element;

while (1) {

choice = menu();

switch (choice) {

case 1:

printf("Enter element to insert: ");

scanf("%d", &element);

root = insert(root, element);

printf("Element %d inserted.\n", element);

break;

case 2:

printf("Enter element to delete: ");

scanf("%d", &element);

root = deleteNode(root, element);

printf("Element %d deleted (if it existed).\n", element);

break;

case 3:

printf("In-order traversal: ");

inOrder(root);

printf("\n");

break;

case 4:

printf("Pre-order traversal: ");

preOrder(root);

printf("\n");

break;

case 5:

printf("Post-order traversal: ");

postOrder(root);

printf("\n");

break;

case 6:

printf("Exiting program.\n");

exit(0);

default:

printf("Invalid choice. Please try again.\n");

}

}

return 0;

}

OUTPUT:

PS D:\C Data Structure\11. Binary Search Tree> gcc binarysearchtree.c

PS D:\C Data Structure\11. Binary Search Tree> ./a

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 1

Enter element to insert: 13

Element 13 inserted.

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 1

Enter element to insert: 36

Element 36 inserted.

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 1

Enter element to insert: 69

Element 69 inserted.

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 3

In-order traversal: 13 36 69

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 4

Pre-order traversal: 13 36 69

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 5

Post-order traversal: 69 36 13

Binary Search Tree Menu

1. Insert Element

2. Delete Element

3. In-order Traversal

4. Pre-order Traversal

5. Post-order Traversal

6. Exit

Enter your choice: 6

Exiting program.

PS D:\C Data Structure\11. Binary Search Tree>