// c program for insertion and deletion of BST.

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

struct node{

int data;

struct node \*left;

struct node \*right;

};

struct node \*root= NULL;

struct node\* createNode(int data){

struct node newNode = (struct node)malloc(sizeof(struct node));

newNode->data= data;

newNode->left = NULL;

newNode->right = NULL;

return newNode;

}

void insert(int data) {

struct node \*newNode = createNode(data);

if(root == NULL){

root = newNode;

return;

}

else {

struct node \*current = root, \*parent = NULL;

while(true) {

parent = current;

if(data < current->data) {

current = current->left;

if(current == NULL) {

parent->left = newNode;

return;

}

}

else {

current = current->right;

if(current == NULL) {

parent->right = newNode;

return;

}

}

}

}

}

struct node\* minNode(struct node \*root) {

if (root->left != NULL)

return minNode(root->left);

else

return root;

}

struct node\* deleteNode(struct node \*node, int value)

{

if(node == NULL){

return NULL;

}

else {

if(value < node->data)

{

node->left = deleteNode(node->left, value);

}

else if(value > node->data)

{

node->right = deleteNode(node->right, value);

}

else if(node->left == NULL && node->right == NULL)

{

node = NULL;

}

else if(node->left == NULL)

{

node = node->right;

}

else if(node->right == NULL)

{

node = node->left;

}

else {

struct node \*temp = minNode(node->right);

node->data = temp->data;

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

}

}

return node;

}

void inorderTraversal(struct node \*node) {

if(root == NULL){

printf("Tree is empty\n");

return;

}

else {

if(node->left!= NULL)

inorderTraversal(node->left);

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

if(node->right!= NULL)

inorderTraversal(node->right);

}

}

int main()

{

insert(5);

insert(3);

insert(7);

insert(6);

insert(1);

insert(9);

printf("Binary search tree after insertion: \n");

inorderTraversal(root);

struct node \*deletedNode = NULL;

deletedNode = deleteNode(root, 9);

printf("\nBinary search tree after deleting node 9: \n");

inorderTraversal(root);

deletedNode = deleteNode(root, 1);

printf("\nBinary search tree after deleting node 1: \n");

inorderTraversal(root);

deletedNode = deleteNode(root, 6);

printf("\nBinary search tree after deleting node 6: \n");

inorderTraversal(root);

return 0;

}

Output:

Binary search tree after insertion:

5 3 7 6 1 9

Binary search tree after deleting 9:

5 3 7 6 1

Binary search after deleting 1:

5 3 7 6

Binary search tree after deleting 6:

5 3 7