DSA Lab 5

Selvakumar – 22MAI1004

**Perform insertion and deletion in BInary Search Tree**

// Insertion and Deletion in Binary Search Tree

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

#define MAX 100

struct Node

{

int data;

struct Node \*left, \*right;

};

struct Node \*newNode(int item)

{

struct Node \*temp = (struct Node \*)malloc(sizeof(struct Node));

temp->data = item;

temp->left = temp->right = NULL;

return temp;

}

void inorder(struct Node \*root)

{

if (root != NULL)

{

inorder(root->left);

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

inorder(root->right);

}

}

struct Node \*insert(struct Node \*node, int data)

{

if (node == NULL)

{

return newNode(data);

}

if (data < node->data)

{

node->left = insert(node->left, data);

}

else if (data > node->data)

{

node->right = insert(node->right, data);

}

return node;

}

struct Node \*minValueNode(struct Node \*node)

{

struct Node \*current = node;

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

{

current = current->left;

}

return current;

}

struct Node \*deleteNode(struct 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)

{

struct Node \*temp = root->right;

free(root);

return temp;

}

else if (root->right == NULL)

{

struct Node \*temp = root->left;

free(root);

return temp;

}

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

root->data = temp->data;

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

}

return root;

}

int main()

{

struct Node \*root = NULL;

int n, x;

printf("Enter the number of elements: ");

scanf("%d", &n);

printf("Enter the elements: ");

for (int i = 0; i < n; i++)

{

scanf("%d", &x);

root = insert(root, x);

}

printf("Inorder traversal of the given tree: ");

inorder(root);

printf("Enter the element to be deleted: ");

scanf("%d", &x);

root = deleteNode(root, x);

printf("Inorder traversal of the modified tree: ");

inorder(root);

return 0;

}

**Output**

**Text

Description automatically generated**