

Veri Yapıları Ödevi

Ad: Ali Can

Soyad: Gündüz

Numara: 202107012102

Ders: Veri Yapıları



[2022]

#include <iostream>

using namespace std;

struct node {

int data;

struct node\* left;

struct node\* right; };

struct node\* dugumAra(struct node\* root, int key){

if(key == root->data) { // eger aradigimiz deger root ise rootu donder

return root; }

else if(key > root->data ) { // eger deger buyukse rootun saginda recursive olarak ara

if(root->right == NULL)

return NULL;

else

return dugumAra(root->right, key); }

else {

if(root->left == NULL) // eger deger kucukse rootun solunda recursive olarak ara

return NULL;

else

return dugumAra(root->left, key); }

}

struct node\* enKucuguGetir(struct node\* current) {

while (current->left != NULL) {

current = current->left; }

return current;

}

void dugumSilme(struct node\* root, int key) {

struct node\* parent = NULL;

struct node\* current = root;

// agacta dolasarak key degerine sahip dugumu ariyoruz

while (current != NULL && current->data != key) {

parent = current;

if (key < current->data)

current = current->left;

else

current = current->right;

}

if (current == NULL)

return;

if (current->left == NULL && current->right == NULL) {

// eger silinecek dalin cocugu yoksa yaprak ise silme islemini yap

if (current != root) {

if (parent->left == current)

parent->left = NULL;

else

parent->right = NULL;

}

else

root = NULL;

delete current;

}

else if (current->left && current->right) {

// eger silinecek dalin iki cocugu varsa en sagindaki en kucuk degeri bul

// onu silinecek degerin yerine yaz ve o degeri sil

struct node\* successor = enKucuguGetir(current->right);

dugumSilme(root, successor->data);

current->data = successor->data;

}

else {

// eger silinecek dalin bir cocugu varsa o node secilir ve o deger parenta atanir

struct node\* child = (current->left) ? current->left : current->right;

if (current != root) {

if (current = parent->left)

parent->left = child;

else

parent->right = child;

}

else

root = child;

delete current;

}

}

void inOrder(struct node\* root) {

if (root != NULL) {

inOrder(root->left);

cout << root->data << " ";

inOrder(root->right);

}

}

struct node\* kokDugum(int key) {

struct node\* root = new node();

root->data = key;

root->left = NULL;

root->right = NULL;

return root;

}

struct node\* dugumEkle(struct node\* root, int key) {

if (root == NULL)

root = kokDugum(key);

else {

if (key < root->data)

root->left = dugumEkle(root->left, key);

else

root->right = dugumEkle(root->right, key);

}

return root;

}

int main()

{

struct node\* root = NULL;

int sayi;

int i = 0;

while (i < 7)

{

cout << "sayi giriniz: ";

cin >> sayi;

root = dugumEkle(root, sayi);

i++;

}

struct node\* dugum = dugumAra(root, sayi); // en son sayinin dugumu

cout << dugum->data << endl; // dugumun degeri

dugumSilme(root, sayi); // en son sayiyi silme

inOrder(root);

return 0;

}