**Чернівецький національний університет імені Юрія Федьковича**

**Інститут фізико – технічних та комп’ютерних наук**

Лабораторна робота № 8

на тему

**“** **Фундаментальні алгоритми на деревах** **”**

Виконала : студентка

групи № 244(А)

Мельничук А.Г.

Затверджено : доцентом ЧНУ

Угрин Д.І.

Чернівці 2022

**Хід роботи**

**Тема :** фундаментальні алгоритми на деревах.

**Мета роботи :** вивчити способи ефективного зберігання та обробки інформації на прикладі бінарних дерев.

**Завдання до роботи :**

1. Створити й вивести на екран дерево.

2. Здійснити обхід дерева в прямому порядку.

3. Написати функцію, яка знаходить найбільший та найменший елементи дерева.

4. Написати процедуру, яка визначає число входжень заданого елемента в дерево.

5. Написати функцію, яка підраховує суму всіх елементів дерева.

*Код програми :*

#include <bits/stdc++.h>

#include <iostream>

using namespace std;

// A tree node

class Node

{

public:

int data;

Node\* left, \*right;

/\* Constructor that allocates a new

node with the given data and NULL

left and right pointers. \*/

Node(int data)

{

this->data = data;

this->left = NULL;

this->right = NULL;

}

};

// Returns maximum value in a given

// Binary Tree

int findMax(Node\* root)

{

// Base case

if (root == NULL)

return INT\_MIN;

int res = root->data;

int lres = findMax(root->left);

int rres = findMax(root->right);

if (lres > res)

res = lres;

if (rres > res)

res = rres;

return res;

}

int findMin(Node\* root)

{

//code

if (root == NULL)

{

return INT\_MAX;

}

int res = root->data;

int left = findMin(root->left);

int right = findMin(root->right);

if (left < res)

{

res = left;

}

if (right < res)

{

res = right;

}

return res;

}

int countOccurrences(Node\* root)

{

int res = 0;

if (root == data)

res++;

return res;

}

int addBT(Node\* root)

{

if (root == NULL)

return 0;

return (root->data + addBT(root->left) + addBT(root->right));

}

int inOrder(Node\* root)

{

if (root != NULL)

{

inOrder(root->left);

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

inOrder(root->right);

}

}

// Driver Code

int main()

{

Node\* NewRoot = NULL;

Node\* root = new Node(2);

root->left = new Node(7);

root->right = new Node(5);

root->left->right = new Node(6);

root->left->left = new Node(4);

root->left->right->left = new Node(1);

root->left->right->right = new Node(11);

root->right->right = new Node(9);

root->right->right->left = new Node(4);

// Function call

cout << "- - - - - - - - - - - - - - - " << endl;

cout << "All elements tree is " << inOrder(root) << endl;

cout << "- - - - - - - - - - - - - - -" << endl;

cout << "Maximum element is " << findMax(root) << endl;

cout << "Minimum element is " << findMin(root) << endl;

cout << "Count elements 4 is " << countOccurrences(root) << endl;

cout << "Sum all elements is " << addBT(root) << endl;

return 0;

}

*Мій опис виконаного завдання :* як і було вказано в завданні , ми створили та вивели на екран дерево. Здійснили обхід даного дерева та виконали наступні пункти:

1) виводить всі елементи дерева

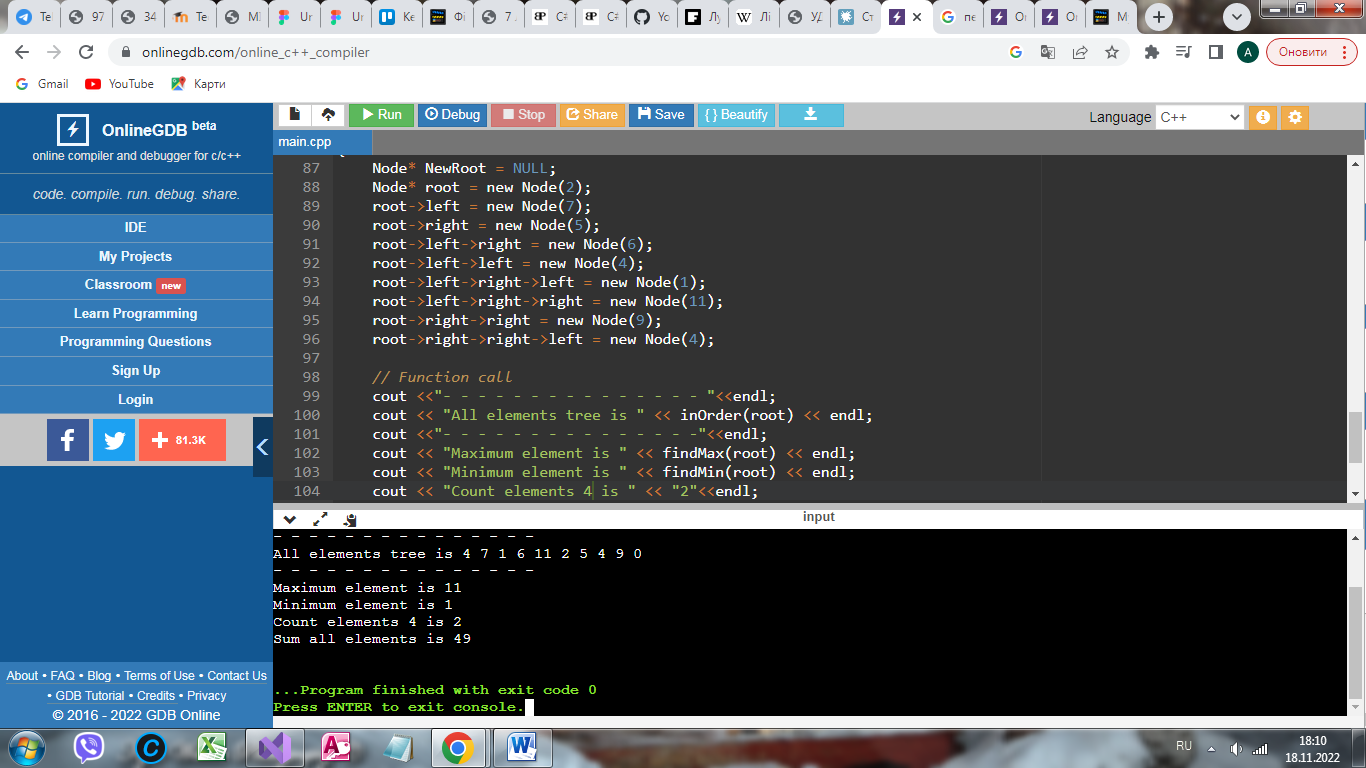
2) шукає мінімальне та максимальне число в даному дереві

3) далі програма шукає кількість входжень заданого числа (в прикладі 4)

4) знаходить суму всіх елементів дерева

5) та виводить всі ці пункти на екран

*Скрін виводу консолі :*



*Висновок :* під час виконання даної лабораторної роботи ми повторили як створювати дерево та як з ним працювати. В даній програмі ми повторили багато операцій над елементами , такі як : пошук мінімального та максимального елементу, пошук суми елементів, пошук кількості входжень даного елементу в масив чисел та інше.