Міністерство освіти і науки України

Національний технічний університет України «Київський політехнічний

інститут імені Ігоря Сікорського"

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 6 з дисципліни

«Основи програмування – 2. Об’єктно-орієнтоване програмування»

« Дерева »

Варіант 21

Виконав студент ІП-11, Ляля Іван Олександрович

Перевірила Вітковська Ірина Іванівна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

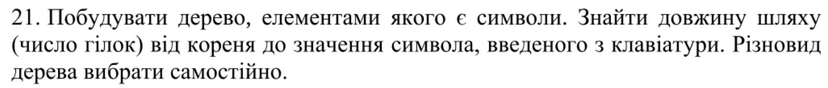
Київ 2022

**Лабораторна робота 6**

**Дерева**

**Мета** – вивчити особливості організації і обробки дерев.

**Задача:**

****

**Постановка задачі:**

Ми описуємо клас Tree\_Node що має атрибути ключ, покажчики на лівий та правий нащадки вузла. Якщо нащадків немає-нульовий покажчик. Дерево по суті буде покажчик на корінь дерева. Робота з деревом буде здійснюватись, рухаючись вже від кореня по гілках.

В головній функції ми створюємо дерево Tree, виводимо його описаним нами методом, знаходимо описаними функцією і методом відстань від кореня до вершини із введеним користувачем ключем. Якщо введеного користувачем ключа немає в дереві – виводимо відповідне повідомлення. Те ж саме робимо й коли в дерево користувач хоче додати вузол з ключем, який вже є в дереві.

**Текст і результат роботи програми на С++:**

**cpp lab 6.cpp:**

#include "Header.h"

int main()

{

setlocale(LC\_ALL, "ukr");

Tree\_Node Tree = Tree\_Fill(); //створюємо і заповнюємо бінарне дерево Tree

cout << "Введене дерево: ";

Tree.Print\_Tree(); //виводимо наше дерево, починаючи від кореня

Distance(Tree); //знаходимо відстань від кореня до введеного символа

return 0;

}

**Header.h:**

#pragma once

#include <iostream>

#include "Tree\_Node.h"

using namespace std;

Tree\_Node Tree\_Fill();

void Distance(Tree\_Node);

**Source.cpp:**

#include "Header.h"

Tree\_Node Tree\_Fill()

{

cout << "Введiть символ-корiнь дерева:";

char s;

cin >> s;

Tree\_Node root(s); //корінь дерева

cout << "Введiть '.' щоб закiнчити введення" << endl;

while (true)

{

cout << "Введiть символ: ";

cin >> s;

if (s != '.') root.insert(s);

else break;

}

return root;

}

void Distance(Tree\_Node root)

{

char s;

cout << "До якого символа шукатимемо вiдстань вiд кореня? -> ";

cin >> s;

if (root.find\_distance(s) != NULL)

cout << "Вiдстань = " << root.find\_distance(s) << endl;

}

**Tree\_Node.h:**

#pragma once

#include <iostream>

using namespace std;

struct Tree\_Node

{

private:

char key;

Tree\_Node\* left, \* right; //покажчики на лівий і правий нащадок вузла відповідно

public:

Tree\_Node(char symb) : key(symb) { left = nullptr; right = nullptr; } //конструктор з параметром-ключем

void insert(char s); //вставка в дерево елемента з ключем s

void Print\_Tree(); //симетричний вивід дерева (або піддерева) з коренем root

int find\_distance(char s, int counter = 0); //знайти кількість гілок між цим вузлом і node2

};

**Tree\_Node.cpp:**

#include "Tree\_Node.h"

void Tree\_Node::insert(char s)

{

if (s < key)

{

if (left == nullptr) left = new Tree\_Node(s);

else (\*left).insert(s);

}

else if (s > key)

{

if (right == nullptr) right = new Tree\_Node(s);

else (\*right).insert(s);

}

else cout << "Елемент з таким ключем уже є в деревi" << endl;

}

void Tree\_Node::Print\_Tree()

{

if (left != nullptr) (\*left).Print\_Tree();

cout << key << ' ';

if (right != nullptr) (\*right).Print\_Tree();

}

int Tree\_Node::find\_distance(char s, int counter)

{

if (key == s) return counter;

else if (s < key)

{

if (left == nullptr) { cout << "Елемента з таким ключем у деревi немає..." << endl; return NULL; }

else (\*left).find\_distance(s, ++counter);

}

else if (s > key)

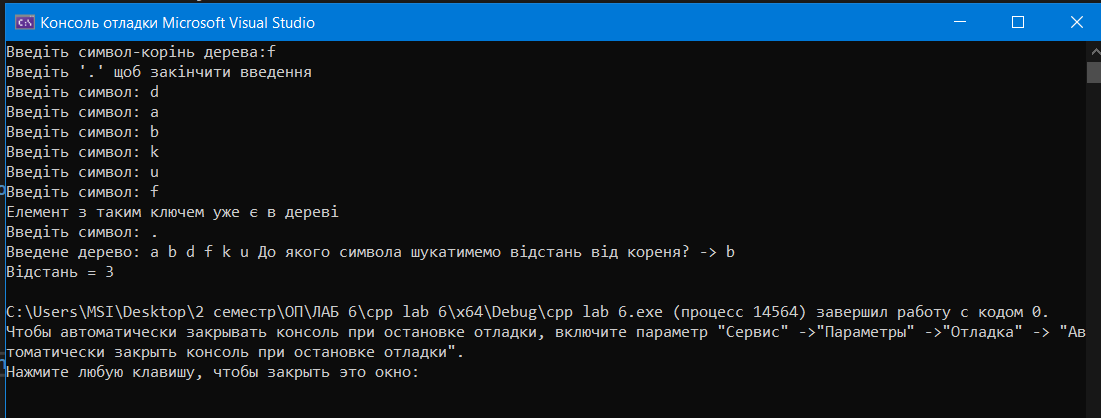
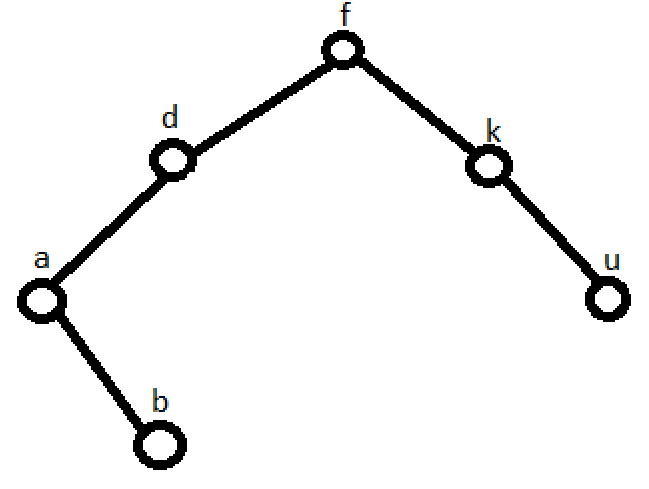
{

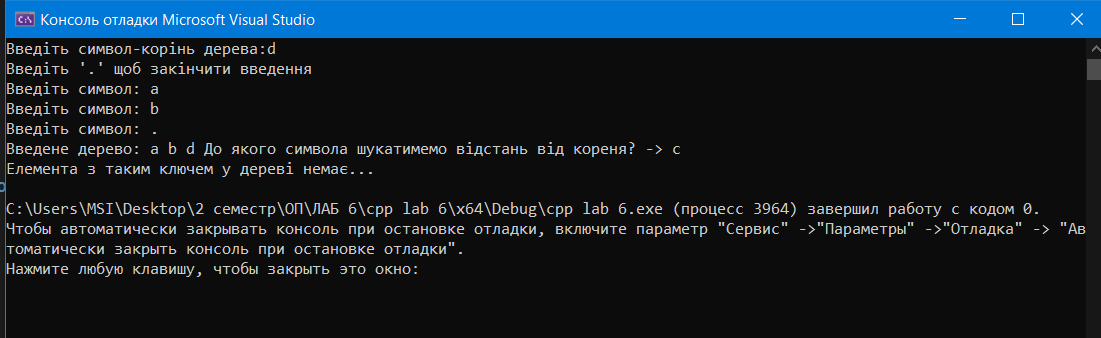
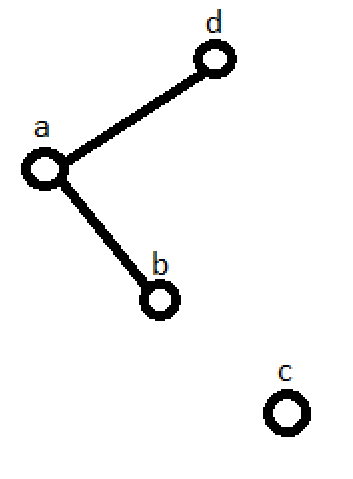
if (right == nullptr) { cout << "Елемента з таким ключем у деревi немає..." << endl; return NULL; }

else (\*right).find\_distance(s, ++counter);

}

}

**** ****

**** ****

**Висновок:**

В ході виконання лабораторного практикуму я навчився особливостей будування і обробки дерев на мові С++. В своєму завданні я запрограмував структуру бінарне дерево, закріпив навички роботи з аналогом класу – структурою, обробив випадки додавання у дерево вузла з уже наявним в дереві ключем та пошуку у дереві такого вузла, якого немає в дереві.