Міністерство освіти і науки України

Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформатика та програмної інженерії

Звіт

З лабораторної роботи № 6 з дисципліни “Основи програмування-2” “Дерева”

Варіант 2

Виконав: ІП-13 Бабашев Олексій

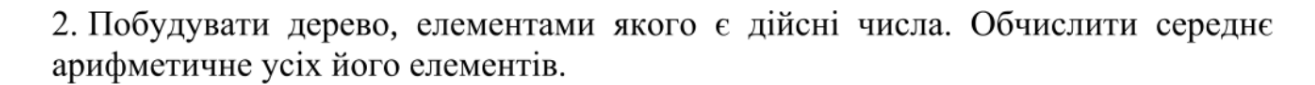
Перевірила: Вєчерковська А. С.

Київ - 2022

**Лабораторна робота 4**

**Класи та об’єкти**

**Індивідуальне завдання:**



**Виконання програми в терміналі та код на С++:**

**Вихідний код:**

**lab6.cpp**

#include"Tree.h"

int main()

{

Tree tree;

cout << "Enter how many numbers you want: ";

int size; cin >> size;

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

{

double val = input();

tree.add\_node(val);

}

tree.print();

cout << endl <<"Average = " << tree.average\_num() << endl;

}

**Tree.cpp**

#include"Tree.h"

Node::Node(double val) {

this->value = val;

right = nullptr;

left = nullptr;

}

Node::~Node() {

delete right;

delete left;

}

Tree::Tree() {

root = nullptr;

}

Tree::~Tree() {

delete root;

}

void Tree::add\_node(double val) {

add\_node\_rec(root, val);

}

void Tree::add\_node\_rec(Node\*& parent, double val) {

if (!parent) {

parent = new Node(val);

parent->left = nullptr;

parent->right = nullptr;

return;

}

else {

if (parent->value < val)

add\_node\_rec(parent->right, val);

else if (parent->value >= val)

add\_node\_rec(parent->left, val);

}

}

void Tree::print() {

print\_rec(root, 0);

}

void Tree::print\_rec(Node\*& parent, int tabs) {

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

{

cout << string(3, ' ') << '|';

}

cout << string(2, '\_');

if (parent != nullptr)

{

cout << parent->value << endl;

print\_rec(parent->right, tabs + 1);

print\_rec(parent->left, tabs + 1);

}

else

cout << endl;

}

double Tree::average\_num() {

double sum = 0;

int num = 0;

average\_num\_rec(root, sum, num);

double avg = sum / (double)num;

return avg;

}

void Tree::average\_num\_rec(Node\*& parent, double& sum, int& num) {

if (parent) {

sum += parent->value;

num++;

if (parent->left)

average\_num\_rec(parent->left, sum, num);

if (parent->right)

average\_num\_rec(parent->right, sum, num);

}

}

double input() {

string s;

double res;

while (true)

{

try

{

cout << "Enter number: "; cin >> s;

res = stod(s);

break;

}

catch (const exception&)

{

cout << "Enter right number!!!!! \n";

}

}

return res;

}

**Tree.h**

#pragma once

#include <iostream>

#include<string>

using namespace std;

class Node {

double value;

Node\* right;

Node\* left;

public:

Node(double);

~Node();

friend class Tree;

};

class Tree {

Node\* root;

void add\_node\_rec(Node\*& parent, double value);

void print\_rec(Node\*& parent, int tabs);

void average\_num\_rec(Node\*& parent, double& sum, int& num);

public:

Tree();

~Tree();

void add\_node(double value);

void print();

double average\_num();

};

double input();

**Висновок:** Вивчив особливості роботи з деревами структури даних та застосував знання на практиці.