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. Модульне програмування»

«Дерева»

Варіант 17

Виконав студент ІП-11 Куценко Артемій Ілліч

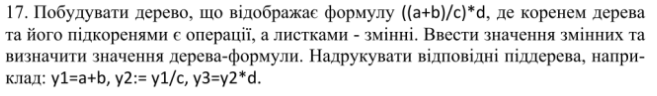
(шифр, прізвище, ім'я, по батькові)

Перевірив Вітковська Ірина Іванівна

(прізвище, ім'я, по батькові)

Київ 2022

**Умова задачі:**



**Код на С++:**

**main.cpp**

#include "pch.h"

int main() {

double a, b, c, d;

MyTree y1; //головне дерево

MyTree y2; //перше піддерево

MyTree y3; //друге піддерево

cout << "Input variables:\n";

cout << "a = "; cin >> a;

cout << "b = "; cin >> b;

cout << "c = "; cin >> c;

cout << "d = "; cin >> d;

cout << '\n';

y3.root = y3.createNode('+');

y3.connectLeft( y3.createNode('a') );

y3.connectRight( y3.createNode('b') );

y2.root = y2.createNode('/');

y2.connectLeft(y3.root);

y2.connectRight(y2.createNode('c'));

y1.root = y1.createNode('\*');

y1.connectLeft(y2.root);

y1.connectRight(y1.createNode('d'));

y1.setValue(a,'a');

y1.setValue(b,'b');

y1.setValue(c,'c');

y1.setValue(d,'d');

y1.calculate();

cout << "\n\n \*\n";

cout << " / d\n";

cout << " + c\n";

cout << "a b\n\n";

y3.printSubtree();

y2.printSubtree();

y1.printSubtree();

}

**Class.h**

#pragma once

#include <iostream>

using namespace std;

struct Node {

Node\* left; // лівий вузол

Node\* right; // правий вузол

char key; // назва змінної або назва операції

double value; // значення змінної або значення результату

Node(char key); // конструктор

double calculate();

Node\* findNode(char key);

};

struct MyTree {

Node\* root;

Node\* createNode(char key);

void connectLeft(Node\* node);

void connectRight(Node\* node);

void setValue(double value, char variableName);

double calculate();

void printSubtree();

private:

Node\* findNode(char key);

};

**Class.cpp**

#include "pch.h"

Node\* MyTree::createNode(char key){

Node\* newNode = new Node(key);

return newNode;

}

void MyTree::connectLeft(Node\* node){

this->root->left = node;

}

void MyTree::connectRight(Node\* node){

this->root->right = node;

}

void MyTree::setValue(double value, char variableName){

Node\* node = this->findNode(variableName);

node->value = value;

}

Node\* MyTree::findNode(char key){

return this->root->findNode(key);

}

double MyTree::calculate(){

return this->root->calculate();

}

void MyTree::printSubtree() {

cout << '\n';

cout << " " << this->root->key << '\n';

cout << this->root->left->value << " " << this->root->right->value;

cout << '\n';

}

Node::Node(char key){

this->left = nullptr;

this->right = nullptr;

this->key = key;

}

double Node::calculate() {

switch (this->key) {

case '+' :

this->value = left->calculate() + right->calculate();

cout << this->value << " = " << left->value << " + " << right->value << '\n';

break;

case '\*' :

this->value = left->calculate() \* right->calculate();

cout << this->value << " = " << left->value << " \* " << right->value << '\n';

break;

case '/' :

this->value = left->calculate() / right->calculate();

cout << this->value << " = " << left->value << " / " << right->value << '\n';

break;

}

return this->value;

}

Node\* Node::findNode(char key) {

if (this->key==key) {

return this;

}

Node\* tempNode = nullptr;

if (left != nullptr) {

tempNode = left->findNode(key);

if (tempNode == nullptr && right != nullptr) {

tempNode = right->findNode(key);

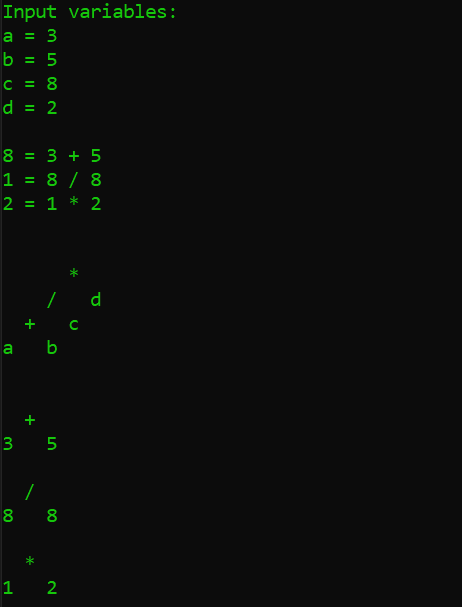
}

}

return tempNode;

}

**Скріншот роботи програми на C++:**



**Висновок:** під час виконання лабораторної роботи я вивчив механізми створення бінарних дерев