МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 12

з навчальної дисципліни

“Базові методології та технології програмування”

ПРОГРАМНА РЕАЛІЗАЦІЯ АБСТРАКТНИХ ТИПІВ ДАНИХ

ВИКОНАВ

студент академічної групи КН-22

Пащенко Данило

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_ Собінов О. Г.

Кропивницький – 2023

**Мета роботи:** полягає у набутті ґрунтовних вмінь і практичних навичок об’єктного аналізу й проектування, створення класів С++ та тестування їх екземплярів, використання препроцесорних директив, макросів і макрооператорів під час реалізації програмних засобів у кросплатформовому середовищі Code::Blocks.

**Завдання до лабораторної роботи**

1. Як складову заголовкового файлу ModulesПрізвище.h розробити клас ClassLab12\_Прізвище –– формальне представлення абстракції сутності предметної області (об’єкта) за варіантом, ― поведінка об’єкта якого реалізовує розв’язування задачі 7.1.

2. Реалізувати додаток Teacher, який видає 100 звукових сигналів і в текстовий файл TestResults.txt записує рядок “Встановлені вимоги порядку виконання лабораторної роботи порушено!”, якщо файл проекта main.срр під час його компіляції знаходився не в \Lab12\prj, інакше –– створює об’єкт класу ClassLab12\_Прізвище із заголовкового файлу ModulesПрізвище.h та виконує його unit-тестування за тест-сьютом(ами) із \Lab12\TestSuite\, протоколюючи результати тестування в текстовий файл \Lab12\TestSuite\TestResults.txt.

Варіант 1

**Опис классу:**

Класс ClassLab12\_Paschenko. Приватні поля: cubeFacesValue – ребра куба, cubePlanesValue – грані куба, cubeIsEmptyInside – пустий чи не пустий. Відкриті методи: getCubeFacesValue() – отримати значення cubeFacesValue, getCubePlanesValue() – отримати cubePlanesValue, getCubeIsEmptyInside() – отримати cubeIsEmptyInside, variableCubeAttributes() – перевантажена функція, яка змінює атрибути класу, getCubeVolume() – отримати об’єм куба.

**Код ModulesPaschenko.h:**

#ifndef MODULESPASCHENKO\_H\_INCLUDED

#define MODULESPASCHENKO\_H\_INCLUDED

#include <cmath>

class ClassLab12\_Paschenko

{

private:

double cubeFacesValue;

double cubePlanesValue;

bool cubeIsEmptyInside;

public:

ClassLab12\_Paschenko(const double facesValue = 1, const double planesValue = 1, const bool isEmptyInside = true)

{

cubeFacesValue = facesValue;

cubePlanesValue = planesValue;

cubeIsEmptyInside = isEmptyInside;

}

double getCubeFacesValue() const {

return cubeFacesValue;

}

double getCubePlanesValue() const {

return cubePlanesValue;

}

bool getCubeIsEmptyInside() const {

return cubeIsEmptyInside;

}

double getCubeVolume(double face);

void variableCubeAttributes(const double facesValue = 1, const double planesValue = 1, const bool isEmptyInside = true);

};

double ClassLab12\_Paschenko::getCubeVolume(double face)

{

variableCubeAttributes(face);

double volume = pow(cubeFacesValue, 3);

return volume;

}

void ClassLab12\_Paschenko::variableCubeAttributes(const double facesValue, const double planesValue, const bool isEmptyInside)

{

cubeFacesValue = facesValue;

cubePlanesValue = planesValue;

cubeIsEmptyInside = isEmptyInside;

}

std::string TestResult(ClassLab12\_Paschenko Cube, int number, double face)

{

std::string result;

result = "Test Suite №"+std::to\_string(number)+"\nFace: "+std::to\_string(Cube.getCubeFacesValue())+"\nPlanes: "+std::to\_string(Cube.getCubePlanesValue())+"\nIs empty: "+std::to\_string(Cube.getCubeIsEmptyInside())+"\nVolume: "+std::to\_string(face)+"\n";

return result;

}

#endif // MODULESPASCHENKO\_H\_INCLUDED**Опис Teacher:**

Створюю екземпляр класу ClassLab12\_Paschenko під назвою Cube. Далі відкриваю файл з тест – кейсами та на запис результатів тестування. Зчитую дані з файлу до поки не закінчаться, виконую метод класу, який поверне об’єм кубу та запишу результат в файл TestResult.txt.

**Код Teacher:**

#include <iostream>

#include <fstream>

#include "ModulesPaschenko.h"

using namespace std;

int main()

{

ClassLab12\_Paschenko Cube;

ifstream f\_read("C:\\Users\\User\\Documents\\Pashchenko-Danylo-KN22\\lab12\\TestSuite\\TestSuite\_12.txt", ios::in);

if(!f\_read.is\_open())

{

cout << "Помилка відкриття файлу" << endl;

return 0;

}

ofstream f\_write("..\\TestSuite\\TestResult.txt", ios::out);

if(!f\_write.is\_open())

{

cout << "Помилка відкриття файлу" << endl;

return 0;

}

string testSuiteLine;

int number = 1;

while (getline(f\_read, testSuiteLine))

{

double volume = Cube.getCubeVolume(stod(testSuiteLine));

f\_write << TestResult(Cube, number, volume);

number++;

}

f\_write.close();

f\_read.close();

system("pause");

return 0;

}

**Висновок:** набув ґрунтовних вмінь і практичних навичок об’єктного аналізу й проектування, створення класів С++ та тестування їх екземплярів, використання препроцесорних директив, макросів і макрооператорів під час реалізації програмних засобів у кросплатформовому середовищі Code::Blocks. Повторив структуру написання програм на мові програмування С++ та структуру модульної програми. Згадав об’явлення функції в мові програмування С++ та з’єднання функцій між собою. Навчився працювати з бібліотекою fstream та її класами ifstream, та ofstream. Дізнався про режими роботи з файлами, такими як: читання, запис (перезапис), дозапис. Та яким чином можна зчитувати інформацію з файла та записувати в файл. Також дізнався, як можно отримати поточний час, за допомогою функції time, як отримати рандомне число, за допомогою функцій srand та rand, також як швидко перетворити ціле число з десяткової системи числення в двійкову систему числення, за допомогою класу bitset. Розробив тест-сьют до програм, згадавши їх структуру та основні складові, та що таке системне тестування. Згадав як їх синтезувати та складати алгоритм розробки програми. Повторив створення власних бібліотеки. Створювати для них функції та створювати заголовковий файл бібліотеки. Навчився підтягувати бібліотеку до проекту в Code::Blocks, налаштовувати Compiler та Linker для проекту, щоб бібліотека запрацювала. Дізнався як створювати Tesr Driver для тестування модулів з бібліотек (Unit Testing). Повторив як працювати з git та git bush для пушінгу файлів в репозиторій git та комітети ці файли. Пригадав ДСТУ 3008:2015. Лабораторна робота сподобалась, було цікаво її виконувати, зауважень к завданню ніяких немає.