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 \_\_\_\_\_

\_\_\_\_\_\_\_\_\_ Барамба А.А

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_ Павло УСІК

Кропивницький – 2022

**ЛАБОРАТОРНА РОБОТА №12**

**ТЕМА:** програмна реалізація абстрактних типів даних

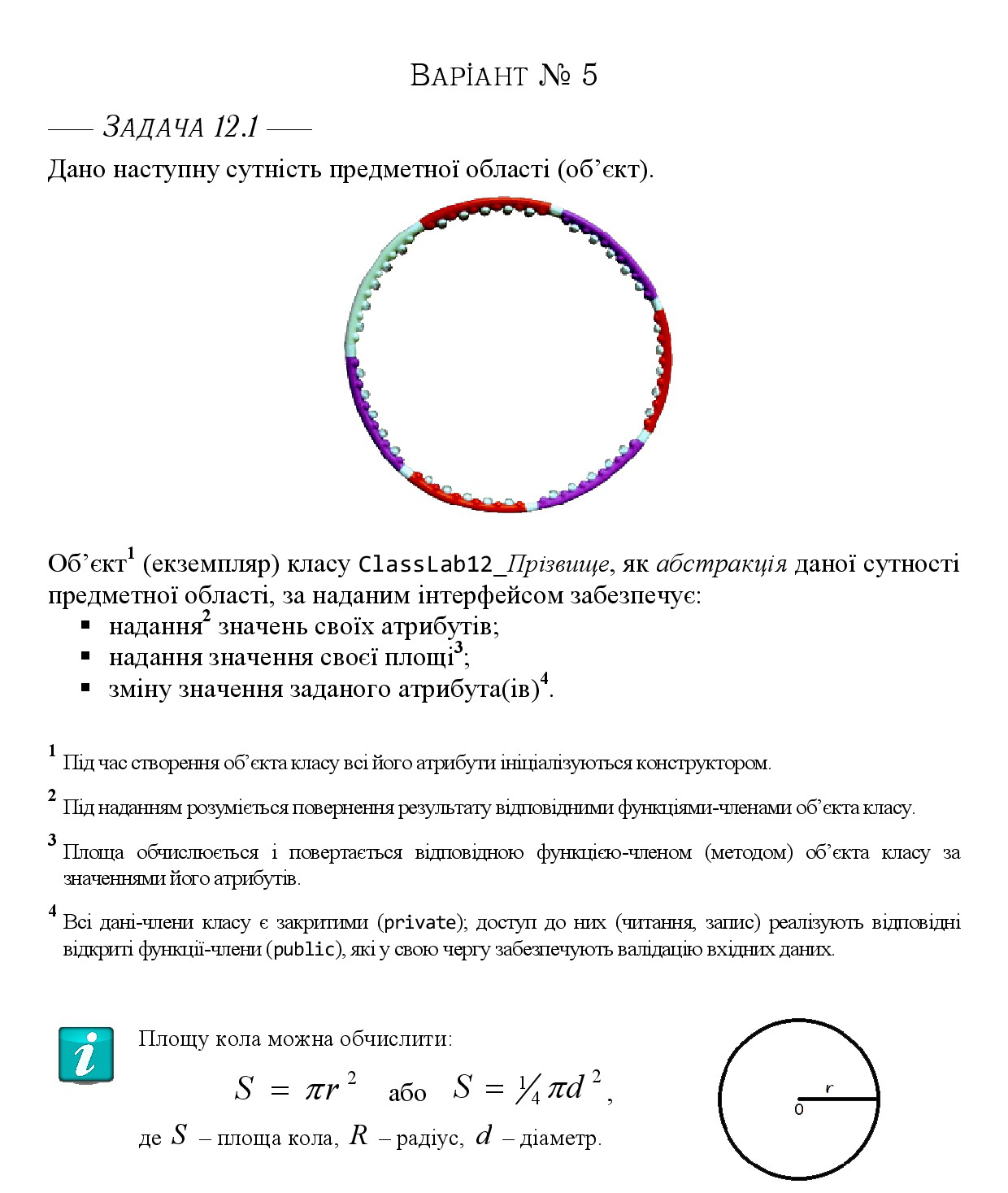
**МЕТА:** полягає у набутті ґрунтовних вмінь і практичних навичок об’єктного аналізу й проектування, створення класів С++ та тестування їх екземплярів, використання препроцесорних директив, макросів і макрооператорів під час реалізації програмних засобів у кросплатформовому середовищі Code::Blocks.

**ВАРІАНТ 5**

**ЗАВДАННЯ:**

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

**ХІД РОБОТИ**

**Завдання 12.1**

**Концептуалізація предметної області:**

За умовою завдання, дано візуалізацію об’єкта, структуру якого можна інтерпретувати як “обруч”. Однак, спираючись на вимоги щодо значень множини вхідних та вихідних даних, достатньо обмежитись концепцією більш абстрактного об’єкта - геометричної фігури коло.

Тому, визначення всіх властивостей сутності описується лише значенням радіуса. Обмеження та допущення щодо вхідних даних зводяться до пошуку найменшої структури, схожою з обраною концепцією, адже маніпулювати довжинами у від’ємних межах - неможливо.

**Вимоги до програмного модуля:**

Властивості об’єкта (радіус – в якості поля) оголошуються на рівні private. Методи отримання значень з рівня private (функції мають префікс get в ідентифікаторі) та методи ініціалізації поля класу (функція, що мають префікс set в ідентифікаторі) оголошуються на рівні public.

Поля класу за замовчуванням ініціалізуються 25.0 (в метричній системі), що надалі використовуються як обмеження або виявлення некоректної ініціалізації

**Тестові артефакти:**

Вхідні значення та очікувані записані в текстовому файлі й приведені до спільної структури, що надалі буде використовуватись додатком Teacher.exe. Протоколювання читання здійснюється 2-ма умовами: у випадку, якщо зчитування неможливе, до текстового файлу TestResults.txt записується “Встановлені вимоги порядку виконання лабораторної роботи порушено!” та завершується робота застосунку; якщо зчитування вдале - відбувається Unit-тестування. Вміст файлу TS.txt:

hoopRadius: 25

hoopSquare: 1963.495408

hoopRadius: 0

hoopSquare: 1963.495408

hoopRadius: -10

hoopSquare: 1963.495408

hoopRadius: 31

hoopSquare: 3019.070540

hoopRadius: 28.71

hoopSquare: 2589.502041

hoopRadius: 28,71

hoopSquare: 2589.502041

**Аналіз та архітектура завдання 12.2:**

На початку роботи застосунку, виконується перевірка, зазначена у вимогах завдання 12.2. Робота застосунку припиняється, якщо файл main.cpp не знайдено в теці ../lab12/prj.

Створюється об’єкт класу ClassLab12\_Baramba, файл TS.txt відкривається для зчитування, файл TestResults.txt відкривається для дозапису (попередній вміст очищається, при відкритті файлу). Відбувається перевірка на наявність даних текстових файлів в директорії та чи відкриті вони на момент роботи даної функції. Якщо хоча б одна умова хибна - робота застосунку припиняться.

Цикл на 10 ітерацій, призначений для декларування процесу тестування, виконує наступний порядок дій:

- оголошення рядків, призначених для збереження радіусу та площі кола. Буферний рядок, що використовуються в проміжних операціях. Змінна типу float, що не має бути ініціалізована.

- зчитування з TS.txt в рядок bufLine методом getline(), починаючи з 13 символа в рядку. Відбуваєтьс окремо для радіуса та площі кола.

- радіус та висота ініціалізуються публічними методами, а площа що отримана публічним методом та приведена до типу string записується в змінну bufLine.

- виконується форматоване виведення значень та отриманого результату Тест Кейсів у файл TestResults.txt.

**ВИСНОВКИ**

Під час виконання лабораторної роботи № 12 на тему «Програмна реалізація абстрактних типів даних» з предмету «Базові Методології та Технології Програмування» я 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.

Під час процесу виконання завдань, виникли наступні спостереження щодо потреб лабораторної роботи: не дивлячись на кінцевий результат реалізації кожного методу та процедури, що не виділяються достатньою складністю, метою лабораторної роботи стала демонстрація навичок коректної абстракції предметної сутності. Розглядаючи точніше - абстрагування мало бути відповідним до вимог задачі, тобто мати обмеження щодо рівня абстрагування (деталізація об’єкта була мінімальною).

Згідно з виданого завдання, абстракція сутності обруча (власний висновок на основі зображення об’єкта) було зведено до простої геометричної фігури - кола. Змінних радіуса та метода отримання площі було достатньо для представлення властивостей кола.

Для кожного поля радіуса відведено два метода, визначені порядком лабораторної: функція отримання значення з приватного поля (метод повертає вміст змінної) та функція ініціалізації даних поля.

Обраною структурою TestSuite став текстовий файл TS.txt, в кожному рядку якого, стала кількість перших символів відведена для дескрипції значення, записаного далі. Таким чином, при зчитуванні даних з файлу, з рядка відкидаються перші 12 символів. Зчитування з TS.txt відбувається в рядок bufLine методом getline(), починаючи з 13 символа в рядку. Відбувається окремо для радіуса та площі кола.

Для зручної навігації та аналізу структури коду, додаткові локальні функції описано в файлі AdditionFunctions.h та перед функцією main у застосунку Teacher.

Реалізовані наступні додаткові функції:

* declareTestSuiteResults(): отримує всі необхідні аргументи, в якості значення, що використовуються для декларування результату модульного тестування за описаним тест кейсом.
* fileInDirectory(): відповідає за перевірку наявності файлу main.cpp в теці ../prj під час компіляції проекту. Містить реалізацію вимог завдання №2.

Результати тестування було записано у текстовий файл TestResults.txt у вигляді таблиці для зручного сприйняття інформації.

**ДОДАТОК А**

(набір тестових результатів TestResults.txt)

--------------------------------------------------

| TEST CASE 1 |

| (input) Hoop radius: 25 м |

| (output) Hoop square: 1963.495408 м^2 |

| (expected) Hoop square: 1963.495408 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

--------------------------------------------------

| TEST CASE 2 |

| (input) Hoop radius: 0 м |

| (output) Hoop square: 1963.495408 м^2 |

| (expected) Hoop square: 1963.495408 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

--------------------------------------------------

| TEST CASE 3 |

| (input) Hoop radius: -10 м |

| (output) Hoop square: 1963.495408 м^2 |

| (expected) Hoop square: 1963.495408 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

--------------------------------------------------

| TEST CASE 4 |

| (input) Hoop radius: 31 м |

| (output) Hoop square: 3019.070540 м^2 |

| (expected) Hoop square: 3019.070540 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

--------------------------------------------------

| TEST CASE 5 |

| (input) Hoop radius: 28.71 м |

| (output) Hoop square: 2589.502041 м^2 |

| (expected) Hoop square: 2589.502041 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

--------------------------------------------------

| TEST CASE 6 |

| (input) Hoop radius: 28 м |

| (output) Hoop square: 2589.502041 м^2 |

| (expected) Hoop square: 2589.502041 м^2 |

| TEST CASE RESULT: true |

--------------------------------------------------

**ДОДАТОК Б**

(Вихідний код ModulesBaramba.h)

#ifndef MODULESBARAMBA\_H\_INCLUDED

#define MODULESBARAMBA\_H\_INCLUDED

#include <cmath>

#define PI 3.14159265359

class ClassLab12\_Baramba{

public:

ClassLab12\_Baramba(){}

ClassLab12\_Baramba(float radius);

void setHoopRadius(float radius);

float getHoopRadius();

float getHoopSquare();

private:

float radius = 25.0;

};

ClassLab12\_Baramba::ClassLab12\_Baramba(float radius){

this->radius = radius;

}

void ClassLab12\_Baramba::setHoopRadius(float radius){

this->radius = radius;

}

float ClassLab12\_Baramba::getHoopRadius(){

return radius;

}

float ClassLab12\_Baramba::getHoopSquare(){

return (PI \* pow(radius, 2));

}

#endif // MODULESBARAMBA\_H\_INCLUDED

**ДОДАТОК В**

(Вихідний код HelperFunctions.h)

**… /Teacher / HelperFunctions.h**

#ifndef HELPERFUNCTIONS\_H\_INCLUDED

#define HELPERFUNCTIONS\_H\_INCLUDED

#include "ModulesBaramba.h"

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

void declareTestSuiteResults(ofstream &resFile, ClassLab12\_Baramba hoop, string outArea, string expArea, int index)

{

resFile << " -------------------------------------------------- " << endl;

resFile << "| TEST CASE " << setw(2) << index << setw(38) << "|" << endl;

resFile << "| (input) Hoop radius: " << setw(20) << hoop.getHoopRadius() << " ì |" << endl;

resFile << "| (output) Hoop square: " << setw(20) << outArea << " ì^2 |" << endl;

resFile << "| (expected) Hoop square: " << setw(20) << expArea << " ì^2 |" << endl;

resFile << "| TEST CASE RESULT: " << setw(20) << boolalpha << (expArea.compare(outArea) == 0) << " |" << endl;

resFile << " -------------------------------------------------- " << endl;

}

#endif // HELPERFUNCTIONS\_H\_INCLUDED

**… /Teacher / main.cpp**

#include "HelperFunctions.h"

using namespace std;

bool fileInDirectory()

{

string cpp = \_\_FILE\_\_;

size\_t found = cpp.find("\\lab12\\prj");

cout << cpp << endl;

if (found == string::npos) {

for (int i = 0; i < 100; i++) {

cout << "\a";

}

ofstream resFile("../../TestSuite/TestResults.txt");

resFile << "Встановлені вимоги порядку виконання лабораторної роботи порушено!" << endl;

resFile.close();

return false;

}

return true;

}

int main()

{

if (!fileInDirectory()) {

cout << "Встановлені вимоги порядку виконання лабораторної роботи порушено!" << endl;

return -1;

}

ClassLab12\_Baramba hoop;

ifstream caseFile("../../TestSuite/TS.txt");

if(!caseFile.is\_open()){ cout << "TS.txt is not opened" << endl; return -2; };

ofstream resultFile("../../TestSuite/TestResults.txt");

if(!resultFile.is\_open()){ cout << "TestResults.txt is not opened" << endl; return -3; };

for (int i = 1; i < 7; i++) {

string hoopRadius;

string hoopSquare;

string bufLine;

float bufFloat;

getline(caseFile, bufLine);

hoopRadius = bufLine.erase(0, 12);

getline(caseFile, bufLine);

hoopSquare = bufLine.erase(0, 12);

hoop.setHoopRadius(stof(hoopRadius));

declareTestSuiteResults(resultFile, hoop, bufLine, hoopSquare, i);

}

caseFile.close();

resultFile.close();

system("pause");

return 0;

}