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
КБ 22-2

_____ Красніцька Є.А.

ПЕРЕВІРИВ
викладач кафедри кібербезпеки
та програмного забезпечення

_____ Олександр СОБІНОВ

Кропивницький – 2022

Тема: Програмна реалізація абстрактних типів даних.

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

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.cpp 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 9

Задача12.1:

Варіант 9

— Задача 12.1 —

Дано наступну сутність предметної області (об’єкт).

Об’єкт[1] (екземпляр) класу `ClassLab12_`*Прізвище*, як *абстракція* даної сутності предметної області, за наданим інтерфейсом забезпечує:

- надання[2] значень своїх атрибутів;
- надання значення периметрів[3] своїх граней (передня/задня, бокові, дно);
- зміну значення заданого атрибута(ів)[4].

[1] Під час створення об’єкта класу всі його атрибути ініціалізуються конструктором.

[2] Під наданням розуміється повернення результату відповідними функціями-членами об’єкта класу.

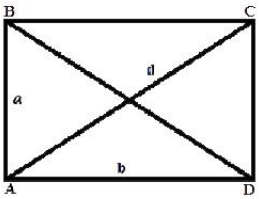
[3] Периметри граней обчислюються і повертаються відповідними функціями-членами (методами) об’єкта класу за значеннями його атрибутів.

[4] Всі дані-члени класу є закритими (`private`); доступ до них (читання, запис) реалізують відповідні відкриті функції-члени (`public`), які у свою чергу забезпечують валідацію вхідних даних.

Периметр прямокутника ABCD дорівнює подвоєній сумі сторін, прилеглих до одного кута:

$$P = 2(a+b),$$

де P – периметр, a, b – довжини сторін прямокутника.



Аналіз задачі 12.1

Зчитувати ми будемо всю інформацію з вхідного файлу, а записувати тільки до вихідного файлу. Ці файли повинні були формату .txt.

Постановка задачі 12.1

Для цієї задачі нам потрібно розробити в .h файлі динамічний об’єкт «Акваріум», який буде мати приватні параметри, такі як: довжина, ширина та висота, також матиме публічні методи за допомогою яких можна задати дані параметрам, або обчислити об’єм.

Аналіз задачі 12.2

Протестувати динамічний об’єкт, який був створено у попередній задачі, й доробити декілька функцій, функції завдання будуть знаходитися у файлі зі класом.

Постановка задачі 12.2

Треба розробити тестовий драйвер, який буде автоматично зчитувати дані з .txt файлу, після чого заносити їх до динамічного списку, а після цього обчислювати їх за допомогою створених методів класу, й записувати результат у вихідний файл.

Додаток А

Лістинг задач 12.1:

```
#ifndef MODULESKRASNITSKA_H_INCLUDED
#define MODULESKRASNITSKA_H_INCLUDED
#include <fstream>
#include <string>

void HelloWorld()
{
    std::cout << "Hello wolrd";
}

class ClassLab12_Krasnitska
{
public:
    ClassLab12_Krasnitska(const float a = 1, const float b= 1):
        a(a),
        b(b),

    {

    }
    float getP();
    float geta();
    float getb ();
    float changea(float a);
```

```
    float changeb(float b);

private:
    float a;
    float b;

};

float ClassLab12_Krasnitska::getP()
{
    return 2*(a+b);
}

float ClassLab12_Krasnitska::changeLength(float a)
{
    a = a;
}

float ClassLab12_Krasnitska::changeWidth(float b)
{
    b = b;
}


float ClassLab12_Krasnitska::getLength()
{
    return a;
}

float ClassLab12_Krasnitska::getHeight()
{
    return b;
}


bool checkFileInFolder()
{
    std::string file = __FILE__;
    size_t checking = file.find("\\prj");

    if (checking == std::string::npos)
    {
        std::ofstream fileResult("../../TestSuite/TestResult.txt");

        fileResult << "Âñòàíîâëåí³ âèìîãè ïîðÿäêó âèêîíàííÿ ëàáîðàòîðíî¿ ðîáîòè
ïîðóøåíî!\n";
        for (int i = 0; i < 100; i++)
            std::cout << "\a";

        fileResult.close();
        return false;
    }

    return true;
}

std::string writeResultToFile(ClassLab12_Krasnitska akvarium, int counter)
{
```

```
    return "Test Suite ¹" + std::to_string(counter)
     + "\nLength = " + std::to_string(akvarium.geta())
     + "\theight = " + std::to_string(akvarium.getb())
     + "\nResult = " + std::to_string(akvarium.getP()) + "\n";
}


#endif // MODULESKRASNITSKA_H_INCLUDED
```

Лістинг Teacher main.cpp:

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include "ModulesKrasnitska.h"

using namespace std;

int main()
{
    string line;
    if(checkFileInFolder() == false)
        return 0;

    ClassLab12_Krasnitska akvarium;
    ifstream Tests_File("../../TestSuite/Tests.txt");
    ofstream TestResult_File("../../TestSuite/TestResult.txt");

    if (Tests_File.is_open() || TestResult_File.is_open())
    {
        int counter = 1;
        float number;
        while (getline(Tests_File, line, '\n'))
        {
            istringstream iss(line);
            vector<float> arrNumbers;

            while (iss >> number)
                arrNumbers.push_back(number);

            akvarium.changeLength(arrNumbers[0]);
            akvarium.changeWidth(arrNumbers[1]);
            akvarium.changeHeight(arrNumbers[2]);
            TestResult_File << writeResultToFile(akvarium, counter);
            counter++;
        }
    }
    Tests_File.close();
    TestResult_File.close();
    system("pause");
    return 0;
}
```

Висновок: Під час виконання лабораторної роботи № 12 на тему «Програмна реалізація абстрактних типів даних» з предмету «Базові Методології та Технології Програмування» я відпрацювала на практиці роботу з класами та засвоїла тестування динамічних типів даних за допомогою модульного тестування. Набула ґрунтовних вмінь і практичних навичок реалізації у Code::Blocks IDE мовою програмування С++ програмних модулів створення й оброблення даних типів масив, структура, об'єднання, множина, перелік, перетворення типів даних, використання файлових потоків та функцій стандартних бібліотек для оброблення символьної інформації. Під час лабораторної роботи було виконано такі пункти:

- Завантажено завдання №12 із репозиторію за допомогою команди git pull
- Після цього було створено папки prj, Software, Test Suite, та Report і заповнено README.md файл.

- Після цього було відправлено ці дані за допомогою команд :

  git add -A

  git commit -m "Add folder prj, Software, Test Suite, Report and feeling README file"

  git push

- Почергово було виконано аналіз і постановку задач 12.1 і 12.2, аналіз вимог до ПЗ та вмісту вхідного текстового файлу, проектування архітектури, детальне проектування програмних модулів; одержані артефакти задокументувати й включити до звіту
- Далі було розроблено один тест-сьют задля проведення автоматизованого unit-тестування заголовкового файлу розв'язування задачі 12.1. Тестові артефакти було задокументовано та включено до звіту як ДОДАТОК Б
- За допомогою отриманих під час проектування програмних модулів артефактами виконати конструювання функцій: мовою програмування С++ було реалізовано функції, які реалізовують розв'язування задач 12.1 і 12.2.
- Скомпільовано проект заголовкового фалу Teacher та завантажено на git repository
- Потім мовою програмування С++ було реалізувано консольний застосунок – тестовий драйвер для модульного тестування функцій розв'язування задач 12.1 за допомогою розробленого тест-сьюту з \Lab12\TestSuite та вхідного і/або вихідного текстового файлу

- Створений застосунок Teacher.exe переміщено у \Lab10\Teacher

  - За допомогою Teacher.exe було виконано автоматизоване тестування розроблених функцій розв’язування задач 12.1 та 12.2.

- Вихідний код (текст) проектів Teacher та ModulesKrasnitska.h включено в звіт як ДОДАТОК А .

В ході лабораторної роботи було на практиці продемонстровано роботу с динамічними об’єктами пам’яті та використано бібліотеки для оброблення текстової інформації. В цілому, робота з класами та тестування їх у C++ є важливими етапами розробки програмного забезпечення, що дозволяють забезпечити правильність та надійність коду.

Додаток Б

Назва тестового набору Test Suite Description	TestSuite
Назва проекта / ПЗ Name of Project / Software	Teacher.exe
Рівень тестування Level of Testing	модульне / Unit Testing
Автор тест-сьюта Test Suite Author	Красніцька Єлизавета Андріївна

Test Case ID	Action	Expected Result	Test Result
TC-01	a = 3, b = 5	P= 16	passed
TC-02	a = 8, b = 2	P = 20	passed
TC-03	a = 1, b = 9	P= 20	passed
TC-04	a = 6, b = 4	P = 20	passed
TC-05	a = 2, b = 7	P = 18	passed
TC-06	a = 5, b = 1	P= 12	passed
TC-07	a = 9, b = 3	P = 24	passed
TC-08	a = 4, b = 6	P= 20	passed
TC-09	a = 7, b = 2	P = 18	passed
TC-10	a = 3, b = 8	P = 22	passed

Додаток В

Зміст вхідного файлу:

15 27

42 18

33 52

61 48

77 25

93 62

55 84

68 97

102 76

88 112

Зміст вихідного файлу:

TestSuit1
a = 15, b = 27
P = 2 * (15 + 27) = 84
TestSuit2
a = 42, b = 18
P = 2 * (42 + 18) = 120
TestSuit3
a = 33, b = 52
P = 2 * (33 + 52) = 170
TestSuit4
a = 61, b = 48
P = 2 * (61 + 48) = 218
TestSuit5
a = 77, b = 25
P = 2 * (77 + 25) = 204
TestSuit6
a = 93, b = 62
P = 2 * (93 + 62) = 310
TestSuit7
a = 55, b = 84
P = 2 * (55 + 84) = 278
TestSuit8
a = 68, b = 97
P = 2 * (68 + 97) = 330
TestSuit9
a = 102, b = 76
P = 2 * (102 + 76) = 356
TestSuit10
a = 88, b = 112
P = 2 * (88 + 112) = 400