Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

# ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8

з навчальної дисципліни “Базові методології та технології програмування”

РЕАЛІЗАЦІЯ СТАТИЧНИХ БІБЛІОТЕК МОДУЛІВ ЛІНІЙНИХ

ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

ВИКОНАВ

студент академічної групи

КІ-21-2 Маленко Андрій

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_  П.С. Усік

Кропивницький – 2022

# Лабораторна робота №8

**Тема:** Реалізація статичних бібліотек модулів лінійних обчислювальних процесів

**Мета:** полягає у набутті ґрунтовних вмінь і практичних навичок застосування теоретичних положень методології модульного програмування, реалізації метода функціональної декомпозиції задач, метода модульного (блочного) тестування, представлення мовою програмування С++ даних скалярних типів, арифметичних і логічних операцій, потокового введення й виведення інформації, розроблення програмних модулів та засобів у кросплатформовому середовищі Code::Blocks (GNU GCC Compiler).

**Варіант 12**

**Завдання:**

1. Реалізувати статичну бібліотеку модулів libModulesПрізвище C/C++, яка містить функцію розв’язування задачі 8.1.

Реалізувати програмне забезпечення розв’язування задачі 8.2 — консольний застосунок

ХІД РОБОТИ



**Строга постановка задачі**

Вхідні дані: x, y, z - дійсні числа;

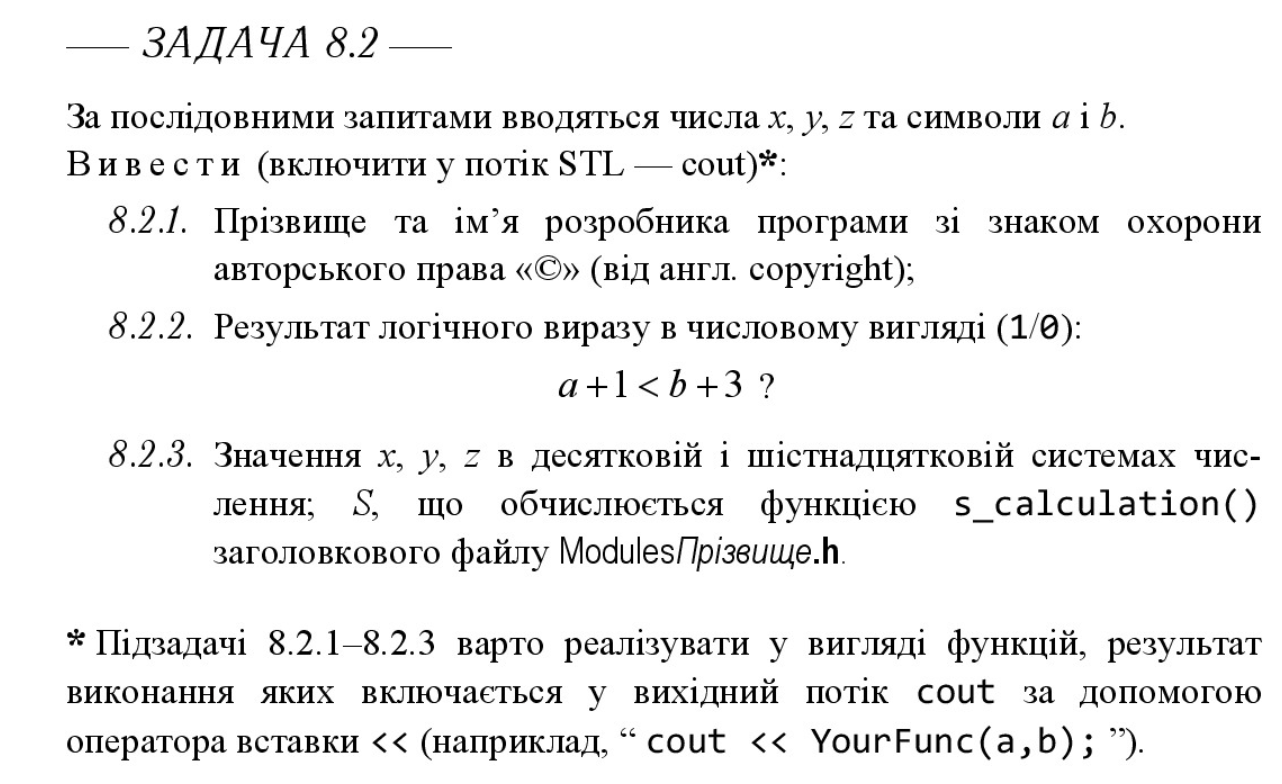
Вихідні дані: S - дійсне число (результат розрахунків);

Проектування програмного модуля:

Модуль ModulesMalenko, що складається з розрахункової функції.

Тестовий драйвер TestDriver - функція, що містить в собі:

* масиви для вхідних значень x, y та z відповідно;
* масив значень очікуваних результатів;
* цикл, для виведення еталонних вхідних значень, очікуваного результату та результату розрахунку модуля ModulesMalenko.



Строга постановка задачі:

Вхідні дані:

* x, y, z - цілочисельні числа;
* a, b – дійсні числа;

Вихідні дані:

* Ім’я розробника з символом copyright;
* Результат логічного виразу: а + 1 < b + 3;
* x, y, z в десятковій та шістнадцятковій системі числення;
* Результат функції s\_calculate з модуля ModulesMalenko;

Процедурна алгоритмізація:

Модуль ModulesMalenko використовуємо з попереднього проекту.

**Copyright()** - виводить прізвище та ім’я розробника, з використанням символу copyright.

**logvir()** - виводить у потік числове логічне значення (1/0) виразу а + 1 < b + 3;

**DecHex()** - слугує для виведення значень x, y, z у десятковій та шістнадцятковій системі числення.

Вищеперераховані функції викликаються в main() після запиту на введення значення певній змінній, а також - до потоку включається функція s\_calculate, що приймає аргументи x, y, z та повертає результат обчислень.

Висновок

Мета цієї роботи полягає у набутті ґрунтовних вмінь і практичних навичок застосування теоретичних положень методології модульного програмування, реалізації метода функціональної декомпозиції задач, метода модульного (блочного) тестування, представлення мовою програмування С++ даних скалярних типів, арифметичних і логічних операцій, потокового введення й виведення інформації, розроблення програмних модулів та засобів у кросплатформовому середовищі Code::Blocks (GNU GCC Compiler).

Файл формату .h називається заголовковим і складається з препроцессорних макропідстановок та оголошених прототипів функцій. #ifndef - макрос, що перевіряє визначеність файлу. Якщо файл не визначено - виконуються #define макропідстановка, що задає файлу визначеності. Внаслідок цього, виконується наступний набір операторів, а в кінці вихідного файлу, перевірка на визначеність файлу припиняється макросом #endif. Інакше кажучи - дані макроси дозволяють, або забороняють компіляцію виділеної ними частини коду.

Файл формату .cpp називається файлом вихідного коду та містить ініціалізацію функцій, раніше оголошених в .h файлі. Потребує підключення директиви .h файлу через макропідстановку #include для належної реалізації функцій.

Якщо підключити модуль через заголовковий файл та спробувати скомпілювати проект, компілятор виведе повідомлення про помилку. Причиною цього є те, що створена статистична бібліотека не є стандартною та компілятор не знає до неї шляху. Для цього треба налатувати опції компілятора Build options: Linker ― вказати шлях до файлу статичної бібліотеки libModulesПрізвище.а, Compiler ― шлях папки, де знаходиться ModulesПрізвище.h. Тільки тоді повідомлень про помилку не буде і можна спокійно компілювати проект.

Робота була виконана за допомогою дистрибутиву C++ та кросплатформованого IDE – Code::blocks.

Також, необхідно виділити увагу бібліотеці <cmath>, що використовувалась для операцій всередині s\_calcuate.

Для використання та подальшого порівняння з очікуваним результатом, розробляємо множину контрольних прикладів (вхідних даних) в якості одновимірного масиву для кожної вхідної змінної. Окремо від функціонуючої програми вирішуємо приклад з задачі 8.1 та створюємо масив очікуваних даних. У вигляді циклу, та виведення повідомлень створюємо модуль системного тестування. Результатом його виконання буде наочне представлення еталонних значень, очікуваних та отриманих результатів у консольному вікні. На основі отриманих та завчасно створених даних формуємо TestSuite.

Завдання 8.2 потребує використання відлагодженого модуля ModulesMalenko. Проект 8.2 користується як процедурною, так і модульною парадигмою програмування (дана потреба викликана умовою задачі). Завдання націлене на детальний огляд роботи потокового виведення за допомогою cout (з простору імен std), функціонування маніпуляторів форматованого виведення та запису виклику функції безпосередньо в потік виведення.

**ДОДАТОК А**

(TestSuite до завдання 8.1)

|  |  |  |
| --- | --- | --- |
|  | Назва тестового набору /  **Test Suite Description** | TS\_8\_1 |
|  | Назва проекта/ПЗ /  **Name of project** | Malenko-task\_8\_1.ехе |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Маленко Андрій |
|  | Виконавець /  **Implementer** | Маленко Андрій |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Actions** | **Expected Result** | **Test Result** |
| **TS\_01** | X = 1  Y = 4  Z = 2 | 3072.665 | Passed |
| **TS\_02** | X = 6  Y = 2  Z = 2 | 95.604 | Passed |
| **TS\_03** | X = 5  Y = 3  Z = 9 | 728.478 | Passed |
| **TS\_04** | X = 0.5  Y = -4.3  Z = 15 | -4410.137 | Passed |
| **TS\_05** | X = -12  Y = -6  Z = 1 | -23327.071 | Passed |

**ДОДАТОК Б**

(TestSuite до завдання 8.2)

|  |  |  |
| --- | --- | --- |
|  | Назва тестового набору /  **Test Suite Description** | TS\_8\_2 |
|  | Назва проекта/ПЗ /  **Name of project** | Malenko\_8\_2.ехе |
|  | Рівень тестування /  **Level of testing** | Системний |
|  | Автор тест-сьюта /  **Test Suite Author** | Маленко Андрій |
|  | Виконавець /  **Implementer** | Маленко Андрій |

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест кейса /  **Test Case ID** | Дії (кроки)  /  **Action (Test Steps)** | Очікуваний результат  /  **Expected Result** | Результат тестування  /  **Test Result** |
| **TS\_01** | 1. Відкрити застосунок | © Маленко Андрій | **Passed** |
| **TS\_02** | 1. Ввести a = 2  2. Ввести b = 6 | Введіть a: 4  Введіть b: 3  Результат логічного виразу  (a + 1) < (b + 3): 1 | **Passed** |
| **TS\_03** | 1. Ввести x = 9  2. Ввести y = 16  3. Ввести z = 42 | x в десятковій: 9 в шістнадцятковій:9  y в десятковій: 16 в шістнадцятковій: 10  z в десятковій: 42 в шістнадцятковій: 2a | **Passed** |
| **TS\_04** | 1. Ввести x = 9  2. Ввести y = 16  3. Ввести z = 42 | Результат виразу: 3.14573e+06 | **Passed** |

**ДОДАТОК В**

(Вихідний код завдання 8.1)

**ModulesMalenko.cbp**

**#**ifndef ModulesMalenko

#define ModulesMalenko

#include <cmath>

double s\_calculation(double x, double y, double z){

return ( sin(x) / sqrt( fabs( (y \* z) / (x + y) ) ) ) + (3 \* pow(y, 5));

}

#endif // MODULESMALENKO\_H\_INCLUDED

**Test.cbp**

#include <iostream>

#include <conio.h>

#include "ModulesMalenko.h"

using namespace std;

int main(){

double valueX[5] = { 1, 6, 5, 0.5, -12 };

double valueY[5] = { 4, 2, 3, -4.3, -6 };

double valueZ[5] = { 2, 2, 9, 15, 1 };

double expectedResult[5] = { 3072.665, 95.604, 728.478, -4410.137, -23327.071 };

string testResult = "Failed";

double result = 0;

for (int i = 0; i < 5; i++) {

cout.precision(3);

cout.setf(std::ios::fixed);

result = floor( s\_calculation(valueX[i], valueY[i], valueZ[i]) \* 1000) / 1000;

if (result == expectedResult[i]) {

testResult = "Passed";

}

cout << "Test case #" << i + 1 << ": " << endl;

cout << "X = " << valueX[i] << endl;

cout << "Y = " << valueY[i] << endl;

cout << "Z = " << valueZ[i] << endl;

cout << "Expected result: " << expectedResult[i] << endl;

cout << "The result obtained: " << result << endl;

cout << "Test result: " << testResult << endl << endl;

testResult = "Failed";

}

getch();

return 0;

}

**ДОДАТОК Г**

(Вихідний код завдання 8.2)

main.cbp

#include <iostream>

#include "ModulesMalenko.h"

using namespace std;

string Copyright(){

return "© Маленко Андрій";

}

bool logvir(float a, float b){

return (a + 1) < (b + 3);

}

void DecHex(int x, int y, int z){

cout << endl;

cout << "x в десятковій: " << dec << x <<" в шістнадцятковій:"<< hex << x << endl;

cout << "y в десятковій: " << dec << y << " в шістнадцятковій: " << hex << y << endl;

cout << "z в десятковій: " << dec << z << " в шістнадцятковій: " << hex << z << endl;

}

int main(){

system("chcp 1251 & cls");

int x = 0;

int y = 0;

int z = 0;

float a = 0;

float b = 0;

cout <<Copyright()<< endl;

cout << "Введіть число a: ";

cin >> a;

cout << "Введіть число b: ";

cin >> b;

cout << "Результат логічного виразу (a + 1) < (b + 3): "

<< logvir(a, b) << endl << endl;

cout << "Введіть число x: ";

cin >> x;

cout << "Введіть число y: ";

cin >> y;

cout << "Введіть число z: ";

cin >> z;

DecHex(x,y,z);

cout << endl << "Результат виразу: " << s\_calculation(x, y, z);

return 0;

}