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-21-1 Ожеховський Владислав

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.
2. Реалізувати програмне забезпечення розв’язування задачі 8.2 — консольний застосунок.

# ХІД РОБОТИ

## C:\Users\vlado\Pictures\Screenshots\Снимок экрана (5).png

Рисунок 1.1 - Умова задачі 8.1

### **Строга постановка задачі:**

Вхідні дані: x, y, z - дійсні числа;

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

### **Проектування програмного модуля:**

Модуль ModulesOzhekhovskyi, що складається з заголовкового файлу (містить оголошення прототипу функції) та файлу вихідного коду (в котрому присутня реалізація функції s\_calculate).

Функція s\_calculate приймає три аргументи в якості вхідних даних, та повертає значення, розраховане за виразом, зазначеним в умові.

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 ModulesOzhekhovskyi.

Вихідний код, див. ДОДАТОК В.

## 

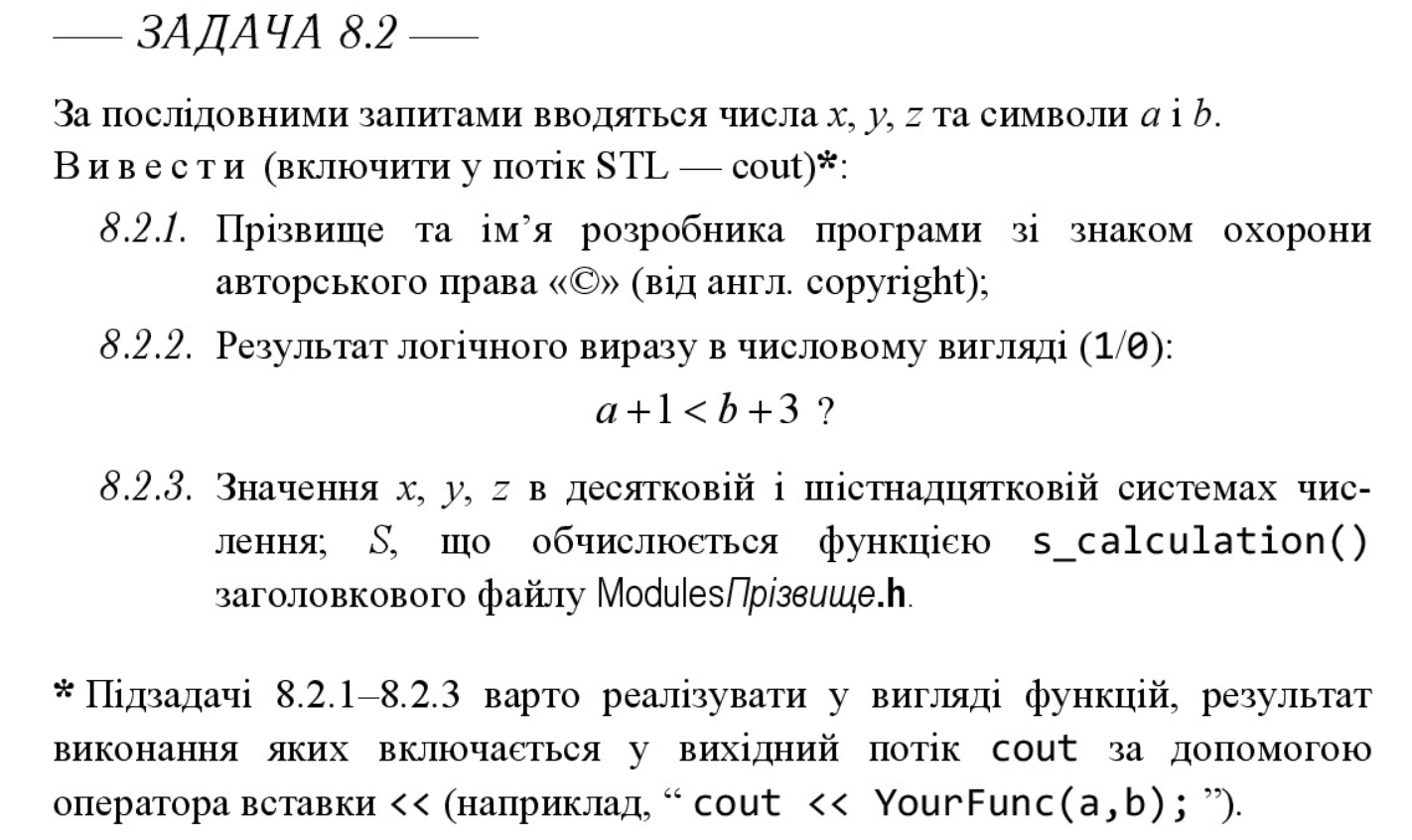


Рисунок 1.2 - Умова задачі 8.2

### **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 ModulesOzhekhovskyi;

### **Процедурна алгоритмізація:**

Модуль ModulesOzhekhovskyi використовуємо з попереднього проекту.

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

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

**showDecimalHexNums()** - 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:

Дана лабораторна робота націлена на набуття навичок у використанні більш упорядкованої парадигми програмування. На відміну від процедурної, де всі функції реалізовані в одному кодовому файлі (що на достатньо масивних проектах викликає ієрархію) - модульна фігурує поняттям “статичної бібліотеки”. Статичні бібліотеки містять в собі інформацію про модулі, що реалізований в якості .h та .cpp файлів.

Заголовковий файл .h формату 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 вихідних кодів формату .cpp містить ініціалізацію функцій, раніше оголошених в .h файлі. Потребує підключення директиви .h файлу через макропідстановку #include для належної реалізації функцій.

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

При виконанні завдань 8.1 та 8.2 був задіяний метод висхідного проектування програмного забезпечення, що полягав у початковій реалізації менш значимих (або допоміжних) функцій нижнього рівня. Такою функцією виявилась s\_calculate, з модуля ModulesOzhekhovskyi, розробленим на етапі задачі 8.1. До даного завдання застосовувались всі правила побудови файлів вихідного коду та заголовкового. Надалі, модуль ModulesOzhekhovskyi використовувався в main функціях обох задач.

Також, необхідно виділити увагу бібліотеці <cmath>, що використовувалась для операцій всередині s\_calcuate. Використовували базові математичні функції pow() – піднесення числа в степінь n, sqrt() – квадратний корінь, fabs() – модуль числа, sin() – синус числа.

Реалізація тестового модуля визначена в main.cpp проекту TestDriver. Основною метою створення даного модуля була автоматизація процесу Unit Testing, що зобов'язує ізоляцію певної процедури всередині цього ж модуля. Це гарантує більш якісне проведення процесу тестування, адже вірогідність виявити помилки та позбутися від них - збільшується.

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.

Результат unit test–у всіх п’ятьох тест кейсів є Passed. Автоматизоване тестування проведено успішно.

Завдання 8.2 потребує використання відлагодженого модуля ModulesOzhekhovskyi. 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** | Ozhekhovskyi-task\_8\_1.ехе |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Ожеховський Владислав |
|  | Виконавець /  **Implementer** | Ожеховський Владислав |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Actions** | **Expected Result** | **Test Result** |
| **TS\_01** | X = 5  Y = 3  Z = 9 | 728.478 | **Passed** |
| **TS\_02** | X = 4.5  Y = 6.3  Z = 1.2 | 29771.927 | **Passed** |
| **TS\_03** | X = 7.8  Y = 2  Z = 3.6 | 97.164 | **Passed** |
| **TS\_04** | X = 0.5  Y = -4.3  Z = 15 | -4410.137 | **Passed** |
| **TS\_05** | X = 11  Y = -3  Z = 2.6 | -730.013 | **Passed** |

**ДОДАТОК Б**

(TestSuite до завдання 8.2)

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

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | © Ожеховський Владислав | **Passed** |
| **TS\_02** | 1. Ввести a = 4  2. Ввести b = 3 | Введіть a: 4  Введіть b: 3  Результат логічного виразу  (a + 1) < (b + 3): 1 | **Passed** |
| **TS\_03** | 1. Ввести x = 8  2. Ввести y = 22  3. Ввести z = 15 | x в десятковій: 8  y в десятковій: 22  z в десятковій: 15 | **Passed** |
| **TS\_04** | 1. Ввести x = 8  2. Ввести y = 22  3. Ввести z = 15 | x в шістнадцятковій: 8  y в шістнадцятковій: 16  z в шістнадцятковій: f | **Passed** |
| **TS\_05** | 1. Ввести x = 8  2. Ввести y = 22  3. Ввести z = 15 | Результат виразу: 1.54609e+07 | **Passed** |

**ДОДАТОК В**

(Вихідний код завдання 8.1)

**… / ModulesOzhekhovskyi / main.cpp**

#include "ModulesOzhekhovskyi.h"

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

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

}

int main (){

return 0;

}

**… / ModulesOzhekhovskyi / ModulesOzhekhovskyi.h**

#ifndef MODULESOZHEKHOVSKYI\_H\_INCLUDED

#define MODULESOZHEKHOVSKYI\_H\_INCLUDED

#include <cmath>

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

#endif // MODULESOZHEKHOVSKYI\_H\_INCLUDED

**… / TestDriver / main.cpp**

#include <iostream>

#include <conio.h>

#include "ModulesOzhekhovskyi.h"

using namespace std;

int main(){

double valueX[5] = { 5, 4.5, 7.8, 0.5, 11 };

double valueY[5] = { 3, 6.3, 2, -4.3, -3 };

double valueZ[5] = { 9, 1.2, 3.6, 15, 2.6 };

double expectedResult[5] = { 728.478, 29771.927, 97.164, -4410.137, -730.013 };

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)

**… / Ozhekhovskyi\_task / main.cpp**

#include <iostream>

#include <windows.h>

#include <conio.h>

#include "ModulesOzhekhovskyi.h"

using namespace std;

string authorCopyright(){

return "© Ожеховський Владислав";

}

bool logicalExpression(double a, double b){

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

}

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

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

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

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

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

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

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

}

int main(){

SetConsoleOutputCP(1251);

int x = 0;

int y = 0;

int z = 0;

double a = 0;

double b = 0;

cout << authorCopyright() << endl;

cout << "Введіть число a: ";

cin >> a;

cout << "Введіть число b: ";

cin >> b;

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

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

cout << "Введіть число x: ";

cin >> x;

cout << "Введіть число y: ";

cin >> y;

cout << "Введіть число z: ";

cin >> z;

showDecimalHexNums(x, y, z);

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

getch();

return 0;

}