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 Назаревський М. В.

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

Усік П. С.

Кропивницький – 2022

# ЛАБОРАТОРНА РОБОТА №8

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

**Мета 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).

## Варіант 11

**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

## 

Рисунок 1.1 - Умова задачі 8.1

### **Строга постановка задачі:**

Вхідні дані: x, y, z - дійсні знакові числа;

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

### **Проектування програмного модуля:**

Модуль ModulesNazarevskyi, що складається з заголовкового файлу та файлу вихідного коду (реалізація функції s\_calculation).

Функція s\_calculation 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 ModulesNazarevskyi.

Для вихідного коду - Додаток В.

## 

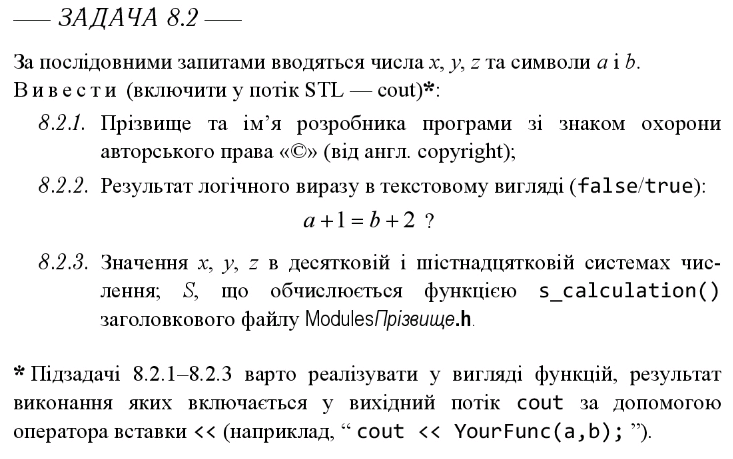


Рисунок 1.2 - Умова задачі 8.2

### **Строга постановка задачі:**

Вхідні дані:

* x, y, z - цілі числа;
* a, b - дійсні числа;

Вихідні дані:

* Ім’я розробника з символом ©;
* Результат логічного виразу: a + 1 = b + 2;
* x, y, z в десятковій та шістнадцятковій системі числення;
* Результат функції s\_calculation з модуля ModulesNazarevskyi;

### **Процедурна алгоритмізація:**

Модуль ModulesNazarevskyi використовуємо з завдання 8.1.

Copyright - виводить прізвище ім’я по-батькові розробника, з використанням символа ©.

inputValues - функція, що приймає 5 аргументів (x, y, z, a, b) та використовує їх для запиту на введення користувачем.

Compare - функція, що порівнює вираз a + 1 = b + 2, та дає точну відповідь true або false

DecAndHex – функція, що слугує для виведення значень x, y, z у десятковій та шістнадцятковій системі числення.

Ці функції викликаються в main, а також - до потоку включається функція s\_calculation що приймає аргументи x, y, z та повертає результат обчислень.

# ВИСНОВКИ

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 реалізовані 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.

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

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 залежності та позбутися від них - збільшується.

Для використання та подальшого порівняння з очікуваним результатом, розробляємо множину контрольних прикладів 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 представлення еталонних значень, очікуваних та отриманих результатів у консольному вікні. На основі отриманих та завчасно створених даних я сформував TestSuite.

Завдання 8.2 потребує використання відлагодженого модуля ModulesNazarevskyi. 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.

Під час завантаження файлів проекту лабаратороної роботи №8 на сайт GitHub.com, я зіткнувся з проблемою відображення української мови (та взагалі кириличних символів) на цьому сайті. Проаналізувавши попередження від GitHub, я припустив, що файлу main.cpp (з Nazarevskyi\_task) потрібно змінити кодировку на UTF-8, для корректного відображення української мови. Після зміни кодировки на UTF-8, я знову завантажив файл main.cpp та проблема була усунена. Цю ж саму дію я зробив для файлу main.cpp (з TestDriver). Кодировку файлу main.cpp з папки ModulesNazarevskyi я не змінював, тому що в ньому я не використовував кириличні символи.

В ДОДАТКУ А указані лістинг TestDriver, UnitTest до функції s\_calculation. В ДОДАТКУ Б указані лістинг Nazarevskyi\_task та Test Suite до завдання 8.2

**ДОДАТОК А**

(UnitTest до завдання 8.1)

|  |  |  |
| --- | --- | --- |
| Artifact: Unit test | Назва тестового набору /  **Test Suite Description** | TS\_8\_1 |
| Date: 03/08/2022 | Назва проекта/ПЗ /  **Name of project** | Назаревський-task\_8\_1.ехе |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Назаревський Максим |
|  | Виконавець /  **Implementer** | Назаревський Максим |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Actions** | **Expected Result** | **Test Result** |
| TS\_01 | X = 3  Y = 2  Z = 5 | -3.05897 | Passed |
| TS\_02 | X = -5  Y = -1  Z = 8 | -7.18740 | Passed |
| TS\_03 | X = 0  Y = 12  Z = 0 | 0 | Passed |
| TS\_04 | X = 11  Y = 5  Z = -6 | -5.72915 | Passed |
| TS\_05 | X = -3  Y = -25  Z = -7 | -6.51101 | Passed |

**Лістинг TestDriver**

#include <iostream>

#include <iomanip>

#include <conio.h>

#include <clocale>

#include "ModulesNazarevskyi.h"

using namespace std;

int main()

{

char \*locale = setlocale(LC\_ALL, "ukr");

float cX[5] = { 3, -5, 0, 11, -3 };

float cY[5] = { 2, -1, 12, 5, -25 };

float cZ[5] = { 5, 8, 0, -6, -7 };

double expectedResult[5] = { -3.05897, -7.1874, 0, -5.72915, -6.51101 };

double doneResult = 0;

double NAC = 0.00005; // ЧИСЛА ПІСЛЯ КОМИ

bool TestResult = false;

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

doneResult = s\_calculation(cX[i], cY[i], cZ[i]);

if (abs(expectedResult[i]) >= abs(doneResult))

{ TestResult = abs(expectedResult[i]) - abs(doneResult) <= NAC; }

else

{ TestResult = abs(doneResult) - abs(expectedResult[i]) <= NAC; }

cout << "Тестові значення №" << i + 1 << " : "

<< "X = " << cX[i]

<< " Y = " << cY[i]

<< " Z = " << cZ[i]

<< endl;

cout << "Очікуваний результат: " << expectedResult[i] << endl;

cout << "Отриманий результат : " << doneResult << endl;

cout << boolalpha << "Тестовий результат : " << TestResult << endl << endl;

}

\_getch();

return 0;

}

**ДОДАТОК Б**

(TestSuite до завдання 8.2)

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TS\_8\_2 |
| Date: 03/08/2022 | Назва проекта/ПЗ /  **Name of project** | Назаревський-task\_8\_2.ехе |
|  | Рівень тестування /  **Level of testing** | Системний |
|  | Автор тест-сьюта /  **Test Suite Author** | Назаревський Максим |
|  | Виконавець /  **Implementer** | Назаревський Максим |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | © Назаревський Максим Володимирович | Passed |
| TS\_02 | 1. Ввести y = 12  2. Ввести x = 35  3. Ввести z = 89  4. Ввести a = 4  5. Ввести b = 3 | Введіть значення y: 12  Введіть значення x: 35  Введіть значення z: 89  Введіть символ a: 4  Введіть символ b: 3  Результат 4 + 1 = 2 + 3 : true  'y' в десятковій: 12  'x' в десятковій: 35  'z' в десятковій: 89  'y' в шістнацятковій: c  'x' в шістнацятковій: 23  'z' в шістнацятковій: 59 | Passed |
| TS\_03 | 1. Ввести y = 12  2. Ввести x = 35  3. Ввести z = 89  4. Ввести a = 2  5. Ввести b = 3  6. Натиснути будь-яку клавішу | …  Результат 2 + 1 = 2 + 3 : false  'y' в десятковій: 12  'x' в десятковій: 35  'z' в десятковій: 89  'y' в шістнацятковій: c  'x' в шістнацятковій: 23  'z' в шістнацятковій: 59  Результат виразу: 68.9002 | Passed |
| TS\_04 | ...  1. Ввести y = 3  2. Ввести x = 8  3. Ввести z = 19  4. Ввести a = 3  5. Ввести b = 3  6. Натиснути будь-яку клавішу | …  'y' в десятковій: 3  'x' в десятковій: 8  'z' в десятковій: 19  'y' в шістнацятковій: 3  'x' в шістнацятковій: 8  'z' в шістнацятковій: 13  Результат виразу: 4.8307 | Passed |
| TS\_05 | 1. Ввести y = 5  2. Ввести x = 12  3. Ввести z = 31  4. Ввести a = 1  5. Ввести b = 2  6. Натиснути будь-яку клавішу | …  Результат виразу: -30.9893 | Passed |

**Лістинг Nazarevskyi\_task**

#include <iostream>

#include <locale.h>

#include <iomanip>

#include <conio.h>

#include <cstring>

#include "ModulesNazarevskyi.h"

using namespace std;

int x = 0, y = 0, z = 0;

char a = '0', b = '0';

string Copyright()

{

return "© Назаревський Максим Володимирович";

}

void inputValues(int Y, int X, int Z, char A, char B)

{

cout << "Введіть значення y: " << endl;

cin >> Y; y = Y;

cout << "Введіть значення x: " << endl;

cin >> X; x = X;

cout << "Введіть значення z: " << endl;

cin >> Z; z = Z;

cout << "Введіть символ a: " << endl;

cin >> A; a = A;

cout << "Введіть символ b: " << endl;

cin >> B; b = B;

}

bool Compare(int A, int B)

{

return A + 1 == B + 2;

}

void DecAndHex(int Y, int X, int Z)

{

cout << "'y' в десятковій: " << dec << Y << endl

<< "'x' в десятковій: " << dec << X << endl

<< "'z' в десятковій: " << dec << Z << endl << endl;

cout << "'y' в шістнацятковій: " << hex << Y << endl

<< "'x' в шістнацятковій: " << hex << X << endl

<< "'z' в шістнацятковій: " << hex << Z << endl;

}

int main()

{

char \*locale = setlocale(LC\_ALL, "ukr");

system("chcp 1251 & cls");

cout << Copyright() << endl << endl;

inputValues(y, x, z, a, b);

cout << boolalpha

<< "Результат: " << a << " + 1 = 2 + " << b << " : " << Compare(a, b) << endl << endl;

DecAndHex(y, x, z);

\_getch();

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

\_getch(); return 0;

}