Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8

з навчальної дисципліни

“Базові методології та технології програмування”

РЕАЛІЗАЦІЯ СТАТИЧНИХ БІБЛІОТЕК МОДУЛІВ

ЛІНІЙНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

[https://github.com/odorenskyi/](https://github.com/odorenskyi/Dmytro-Parkhomenko-KB18)

ВИКОНАВ

студент академічної групи КБ-24

Марченко К. О.

ПЕРЕВІРИВ

викладач кафедри кібербезпеки   
та програмного забезпечення

Коваленко А. С.

Кропивницький – 2025

**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).

**Завдання до лабораторної роботи**:

1. Реалізувати статичну бібліотеку модулів libModulesMarchenko C/C++, яка містить функцію розв’язування задачі 8.1.
2. Реалізуватии програмне забезпечення розв’язування задачі 8.2 — консольний застосунок.

**Аналіз і постановка задачі 8.1:**

Функціональні вимоги

* Вхідні дані: три числових значення – x, y, z.
* Обчислення: обчислення S за формулою
* Вихід: Результат обчислення S, який має бути коректним при виконанні операцій над дійсними числами.

Обмеження і валідація

* Ділення на нуль: Вираз x/|y − z| → вимагає, щоб |y − z| ≠ 0.
* Вираз (x² + z) / √(|z + x|⋅y) → вимагає, щоб вираз під квадратним коренем (|z + x|⋅y) > 0.
* Вираз cos(x⋅z²)/(2y + π) → знаменник не повинен дорівнювати нулю.

Область визначення функції:

* Вираз під коренем має бути невід’ємним, інакше операція квадратного кореня не може бути виконана над дійсними числами.

Результати тестування s\_calculation зі статичної бібліотеки ModulesMarchenko.а тестовим драйвером:

Test case #1: s\_calculation(2, 3, 1) = 2.64687 == 2.64687 --> passed

Test case #2: s\_calculation(2, 3, 3) = nan == nan --> passed

Test case #3: s\_calculation(0, 4, -1) = nan == nan --> passed

Test case #4: s\_calculation(3, 10, 2) = 2.25376 == 2.25376 --> passed

Test case #5: s\_calculation(1, 0, 2) = nan == nan --> passed

Вихідний код проєкту ModulesMarchenko:

#include <cmath>

#include <stdexcept>

const double EPSILON = 1e-9;

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

// Перевірка для виразу |y-z| (щоб уникнути ділення на 0)

if (std::fabs(y - z) < EPSILON) {

return NAN;

}

// Обчислення першого терму: (x/|y-z|)^3

double term1 = std::pow(x / std::fabs(y - z), 3);

// Перевірка для виразу під квадратним коренем у знаменнику другого терму:

// √(|z+x| \* y) - перевіряємо, що y > 0 та |z+x| > 0

if (y <= 0 || std::fabs(z + x) < EPSILON) {

return NAN;

}

double sqrtDenom = std::sqrt(std::fabs(z + x) \* y);

double term2 = (x \* x + z) / sqrtDenom;

// Перевірка знаменника для третього терму: 2\*y + π

double denominator = 2 \* y + M\_PI;

if (std::fabs(denominator) < EPSILON) {

return NAN;

}

double term3 = std::cos(x \* z \* z) / denominator;

// Обчислення виразу під квадратним коренем

double innerExpression = term1 + term2 - term3;

if (innerExpression < 0) {

return NAN;

}

// Обчислення кінцевого значення S

double S = 1 + std::sqrt(innerExpression);

return S;

}

Вихідний код проєкту TestDriver:

#include <iostream>

#include <sstream>

#include <iomanip>

#include <cmath>

#include "ModulesMarchenko.h"

using namespace std;

string doubleToString(double value, int precision = 5) {

ostringstream oss;

oss << fixed << setprecision(precision) << value;

return oss.str();

}

void test\_s\_calculation() {

double x[5] = {2, 2, 0, 3, 1};

double y[5] = {3, 3, 4, 10, 0};

double z[5] = {1, 3, -1, 2, 2};

double expected\_results[5] = {2.64687, NAN, NAN, 2.25376, NAN};

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

double result = s\_calculation(x[i], y[i], z[i]);

string resultStr = doubleToString(result);

string expectedStr = doubleToString(expected\_results[i]);

bool passed = false;

if (std::isnan(result) && std::isnan(expected\_results[i])) {

passed = true;

} else if (resultStr == expectedStr) {

passed = true;

}

cout << "Test case #" << i+1 << ": s\_calculation("

<< x[i] << ", " << y[i] << ", " << z[i] << ") = "

<< resultStr << " == " << expectedStr

<< " --> " << (passed ? "passed" : "failed") << endl;

}

}

int main() {

test\_s\_calculation();

return 0;

}

## Аналіз та постановка задачі 8.2

Задача 8.2 складається з трьох підзадач, які виконуються послідовно після введення чисел та символів:

**8.2.1:** Виведення прізвища та імені розробника з використанням символу «©».

**8.2.2:** Обчислення логічного виразу

з виведенням результату у числовому вигляді (1 або 0).

**8.2.3:** Виведення чисел x, y, z в десятковій та шістнадцятковій системах числення, а також обчислення значення S за допомогою функції s\_calculation(), яка знаходиться у заголовковому файлі ModulesMarchenko.h.

### Функціональні вимоги

* **Вхідні дані:**
  + Три числових значення x, y, z (можуть бути як цілі, так і дійсні).
  + Два символи a та b (їх значення використовуються у логічному виразі).
* **Вивід:**
  + Результат кожної підзадачі має бути виведений у стандартний потік виводу cout за допомогою оператора вставки <<.
  + Результати мають включати:
    1. Прізвище та ім'я розробника з символом «©».
    2. Результат логічного виразу у вигляді 1 (true) або 0 (false).
    3. Значення x, y, z у десятковій та шістнадцятковій формах; обчислене значення S.

### Обмеження та перевірки

* Забезпечити коректність введення числових значень і символів.
* Обчислення абсолютного значення та порівняння з модифікованим значенням b.
* Форматування даних для числових систем (десяткова та шістнадцяткова).

