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/

ВИКОНАВ

студент академічної групи
КБ-24

Олефіров Г.Є.

ПЕРЕВІРИВ

ст. викладач
кафедри кібербезпеки
та програмного
забезпечення

Коваленко А.С.

Кропивницький – 2025

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 libModulesПрізвище C/C++, яка містить функцію розв’язування задачі 8.1.
2. Реалізувати програмне забезпечення розв’язування задачі 8.2 — консольний застосунок.

ЗАВДАННЯ 8.1

Аналіз задачі

Вхідні дані: x, y, z — числові значення (реальні або цілі).

Вихідні дані: S — результат обчислення за заданою формулою.

Додавання та піднесення до квадрата.

Квадратний корінь.

Тригонометрична функція cos(z).

Дробові вирази та вкладені корені.

Постановка задачі

Задача полягає в розробці алгоритму, який:

1. Приймає на вхід три значення x,y,z.
2. Перевіряє коректність введених даних відповідно до вказаних умов.
3. Обчислює значення *S* за заданою формулою.
4. Виводить результат або повідомлення про помилку, якщо вхідні дані не відповідають умовам.

Аналіз вимоги

1.Отримувати вхідні дані x,y,z.

2.Перевіряти коректність вхідних значень:

z≥0 (щоб вираз під коренем був коректним).

y+4(x3+cosz)>0 (щоб не ділити на нуль і мати коректний результат).

3.Обчислювати значення функції:

Лістинг TestDriver

```
#include <iostream>
#include <cmath>
#include "ModulesOlefirov.h"
#include <locale.h>
using namespace std;
int main() {
  setlocale(LC_ALL , "ukr");
```

```
        double x, y, z;
        cout << "Введіть x, y, z: ";
        cin >> x >> y >> z;
        double result = calculate_S(x, y, z);
        if (result == -1)
      {
        cout << "Помилка: недопустимі вхідні дані!" << endl;
      }   else
      {
          cout << "Результат S: " << result << endl;
      }
        return 0;
      }
```

Лістинг ModuleOlefirov

```
#include <iostream>

 #include <cmath>

using namespace std;

// Функція для обчислення S

double calculate_S(double x, double y, double z) {

// Перевірка на некоректне z

if (z < 0) return -1;

// Обчислення знаменника

double denominator = y + 4 * (pow(x, 3) + cos(z));

// Перевірка на ділення на 0 або від’ємне значення в корені
```

```
if (denominator <= 0) return -1;

// Обчислення чисельника

double numerator = 2 * M_PI * sqrt(0.5 * z);

// Рахує значення виразу під коренем

double root_part = sqrt(numerator / denominator);

// Повертає результат виразу

return pow(z + y + z, 2) - root_part; }
```

ЗАВДАННЯ 8.2

Постановка задачі

Користувач вводить три числа x,y,z та два символи a і b.

Потрібно реалізувати три підзадачі:

Вивести ім'я розробника та символ авторського права ©.

Обчислити логічний вираз a+1<b та вивести результат у числовому вигляді (1 або 0).

Вивести числа x,y,z у десятковій та шістнадцятковій системах числення та обчислити значення виразу S використовуючи функцію `s_calculation()` із заголовкового файлу `Modules/Прізвище.h`.

Вимоги до реалізації:

Кожну підзадачу потрібно оформити у вигляді окремої функції.

Всі функції повинні виводити результат у потік **STL cout**.

Функція s_calculation() винесена в окремий заголовковий файл, що дозволяє використовувати її повторно.

Лістинг 8.2

```
#include  <iostream>

#include  <cmath>

#include "ModulesOlefirov.h"

 #include <windows.h>

using namespace std;

// Функція для виводу інформації про автора all rights reserved

void Annotation() { cout
<<"================================================================
=="<< endl; cout <<"| # * # * Developer: Olefirov Glib * # * * #
|"<< endl; cout <<"| * * # Розробник: Олефіров Гліб * # |"<< endl;
cout <<"| # * # * all rights reserved © # * * # |"<< endl; cout
<<"================================================================
=="<< endl; }

// Функція для обчислення логічного виразу bool Logic(char a, char
b) { return (a + 1 < b); // Повертає true або false }

// Функція для обробки чисел та їх виводу

void processNumbers(int x, int y, int z) {
```

```
cout << "x = " << x << " (0x" << hex << x << dec << ")" << endl;
cout << "y = " << y << " (0x" << hex << y << dec << ")" << endl;
cout << "z = " << z << " (0x" << hex << z << dec << ")" << endl;

}

int main() {

 SetConsoleOutputCP(1251);

Annotation();

double x, y, z; char a, b;

// Введення даних користувачем

cout << "Введіть x, y, z: "; cin >> x >> y >> z; cout << "Введіть
символи a i b: "; cin >> a >> b;

// Виконання підзадач

cout << "Результат логічного виразу (1/0): " << Logic(a, b) << endl;
processNumbers(x, y, z);

double result = calculate_S(x, y, z);

if (result == -1) {

cout << "Помилка: недопустимі вхідні дані!" << endl;

}

else

{

 cout << "Результат S: " << result << endl; } return 0;

}
```

Висновки

1. В ході виконання лабораторної роботи було досліджено принципи роботи статичних бібліотек у мовах програмування C/C++.
2. Було створено статичну бібліотеку libModuleA.a, яка містить функцію s_calculation, та підключено її до основного проєкту.
3. Виконано тестування бібліотеки libModuleA.a за допомогою бібліотеки Libtabtest та перевірено коректність обчислень функції s_calculation.
4. Ознайомлено з методами автоматизації тестування за допомогою CMake та TestDriver, що дозволило спростити процес тестування та інтеграції коду.
5. Було вивчено та реалізовано використання системи контролю версій Git, створено репозиторій та проведено комітинг змін.
6. Виконано тестування коду відповідно до вимог стандарту ISO/IEC 12207, що забезпечило відповідність програмного забезпечення встановленим стандартам якості.
7. У результаті проведеного тестування за допомогою TestDriver було отримано підтвердження коректності реалізації всіх компонентів та їх відповідності очікуваним результатам.
8. Вивчено процес компіляції та лінкування статичних бібліотек, що дозволило зрозуміти, як вони використовуються в розробці програмного забезпечення.
9. Виконано компіляцію бібліотеки та програми за допомогою GCC та Code::Blocks, що дало змогу перевірити сумісність із різними інструментами.
10. Реалізовано тестові функції, які дозволили оцінити коректність роботи коду та знайти потенційні помилки.
11. Вдосконалено навички роботи з терміналом та командним рядком для роботи з бібліотеками та тестуванням.

12.Використано принцип модульного програмування, що полегшило розробку та налагодження коду.

13.Вивчено та реалізовано використання заголовкових файлів для інкапсуляції функцій у бібліотеці.

14.Проведено аналіз ефективності використання статичних бібліотек у порівнянні з динамічними.

15.Досліджено способи підключення бібліотек у різних середовищах розробки.

16.Виконано тестування програми на різних наборах вхідних даних для перевірки її стійкості до некоректних значень.

17.Зрозуміло, як працює механізм лінкування бібліотек при компіляції програми.

18.Вивчено принципи написання тест-кейсів та використання автоматизованих засобів тестування.

19.Виконано рефакторинг коду для покращення його читабельності та підтримуваності.

20.Застосовано принципи структурного програмування для покращення логіки програми.

21.Ознайомлено з особливостями роботи компіляторів GCC та MinGW при роботі з бібліотеками.

22.Використано оператори виводу та форматування даних для покращення читабельності результатів роботи програми.

23.Вивчено та застосовано принципи модульного тестування під час розробки бібліотеки.

24.Ознайомлено з форматом і структурою файлів .a та .lib, що використовуються для статичних бібліотек.

25.Вивчено відмінності між статичними та динамічними бібліотеками на практичних прикладах.

26.Використано макроси препроцесора для керування підключенням бібліотек у заголовкових файлах.

27.Впроваджено засоби перевірки граничних умов та виняткових ситуацій у коді.

28.Перевірено коректність роботи логічного виразу у функції evalLogic з різними типами вхідних даних.

29.Ознайомлено з механізмом обчислення виразів із використанням стандартної математичної бібліотеки.

30.Виконано перевірку роботи основних функцій на різних платформах та середовищах розробки.

31.Використано CMake для автоматизації складання проєкту та підключення бібліотек.

32.Проведено аналіз швидкодії та ефективності реалізованих алгоритмів.

33.Ознайомлено з методами налагодження коду та виявлення помилок під час компіляції та виконання.

34.Виконано документування коду для покращення його читабельності та зрозумілості.

35.Реалізовано механізми виводу повідомлень про помилки у разі некоректних вхідних даних.

36.Досліджено, як працює статична бібліотека при використанні в інших проєктах.

37.Виконано оптимізацію коду шляхом усунення зайвих операцій та дублювання коду.

38.Ознайомлено з процесом створення власних бібліотек для подальшого використання у великих проєктах.

39.Вивчено, як працюють інструменти для автоматичного тестування програмного забезпечення.

40.Виконано перевірку роботи програми з великими наборами вхідних даних.

41.Проаналізовано, як зміна параметрів функції впливає на її продуктивність.

42.Ознайомлено з концепцією управління залежностями під час розробки програмного забезпечення.

43.Досліджено, як зміни у коді бібліотеки впливають на роботу підключеного до неї проєкту.

44.Виконано аналіз складності алгоритмів, що використовуються у функціях бібліотеки.

45.Впроваджено практичні навички використання стандартної бібліотеки cmath для виконання математичних операцій.

46.Виконано перевірку коректності перетворення типів у функціях програми.

47.Ознайомлено з механізмами управління пам’яттю при роботі з бібліотеками.

48.Вивчено принципи написання документації для створених бібліотек та її важливість для подальшого використання.

49.Виконано валідацію вхідних даних та обробку помилок під час виконання програми.

50.Отримано практичні навички створення, використання, тестування та документування статичних бібліотек у C++.