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 КН-24
Булюкін В. Ю.
https://github.com/kuroshi-dev

ПЕРЕВІРИВ
ст. викладач кафедри кібербезпеки
та програмного забезпечення
Коваленко А. С.

Кропивницький – 2025

12 Варіант
Тема: Програмна реалізація оброблення масивів даних та символьної інформації.

Мета: Набуття 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 8.1

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

ModulesBuliukin.cpp

```
#include "ModulesBuliukin.h"

using namespace std;

void printDeveloperInfo() {
    cout << "© Розробник: Булюкін В.Ю." << endl;
}
float s_calculation(float x, float y, float z) {
    if (x + y <= 0) {
        throw invalid_argument("Помилка: x + y має бути більше 0.");
    }
    float S = sin(x) + (3 * pow(y, 5)) / sqrt(x + y);
    return S;
}
```

ModulesBuliukin.h

```
#ifndef MODULES_BULIUKIN_H
#define MODULES_BULIUKIN_H

#include <cmath>
#include <stdexcept>
#include <iostream>
#include <iomanip>

void printDeveloperInfo();
float s_calculation(float x, float y, float z);

#endif
```

TestDriver.cpp

```
#include "../ModulesBuliukin/ModulesBuliukin.h"

using namespace std;

void runTests() {
    try {
        float S = s_calculation(1.5, 2.0, 3.0);
        cout << "Тест 1: Успішно. S = " << S << endl;
```

```
    } catch (...) {
        cout << "Тест 1: Провалено" << endl;
    }

    try {
        s_calculation(-2.0, -1.0, 0.0);
        cout << "Тест 2: Провалено" << endl;
    } catch (const invalid_argument& e) {
        cout << "Тест 2: Успішно. Виявлено помилку: " << e.what() << endl;
    }
}

int main() {
    runTests();
    return 0;
}
```

Завдання 8.2

За послідовними запитами вводяться числа x, y, r та символи a і b. Вивести (включити у потік STL - cout)*:

8.2.1. Прізвище та ім'я розробника програми зі знаком охорони авторського права.

8.2.2. Результат логічного виразу в числовому вигляді (1/0): a+1<b+3?

8.2.3. Значення x, y, и в десятковій і шістнадцятковій системах числення; S, що обчислюється функцією s_calculation() заголовкового файлу ModulesBuliukin.h.

```
#include "../ModulesBuliukin/ModulesBuliukin.h"

using namespace std;
void printDeveloperInfo() {
    cout << "© Розробник: Булюкін В.Ю." << endl;
}

bool logicalExpression(char a, char b) {
    return (a + 1) < (b + 3);
}
void printNumbersInDecAndHex(float x, float y, float z) {
    cout << "Десяткова система: x = " << x << ", y = " << y << ", z = " << z << endl;
     cout << hex << showbase << "Шістнадцяткова система: x = " << (int)x << ", y = " <<
(int)y << ", z = " << (int)z << endl;
    cout << dec;
}
int main() {
    printDeveloperInfo();
    float x, y, z;
    char a, b;

    cout << "Введіть два символи (a та b): ";
    cin >> a >> b;
     cout << "Результат логічного виразу (a + 1 < b + 3): " << logicalExpression(a, b)
<< endl;
    cout << "Введіть три числа (x, y, z): ";
    cin >> x >> y >> z;
    printNumbersInDecAndHex(x, y, z);
    try {
        float S = s_calculation(x, y, z);
        cout << "Результат обчислення S: " << fixed << setprecision(2) << S << endl;
    } catch (const invalid_argument& e) {
        cerr << e.what() << endl;
    }
    return 0;
}
```

Висновок

У ході виконання лабораторної роботи було досягнуто таких результатів

1. Реалізовано статичну бібліотеку libModulesBuliukin.a з функцією s_calculation(), яка коректно обчислює заданий математичний вираз з обробкою помилок (ділення на нуль, невизначеність логарифма).
2. Розроблено консольний застосунок, який:
    a. Виводить інформацію про розробника з символом ©
    b. Обчислює логічний вираз a + 1 < b + 3
    c. Конвертує числа у десяткову та шістнадцяткову системи
    d. Використовує функції з бібліотеки для обчислення S
3. Застосовано модульний підхід:
    a. Чітке розділення коду на логічні модулі
    b. Використання заголовочних файлів для інтерфейсів
    c. Створення тестових драйверів для перевірки функціональності
4. Виконано тестування:
    a. Модульне тестування функції s_calculation()
    b. Системне тестування готового застосунку
    c. Покриття всіх граничних випадків
5. Набуті практичні навички:
    a. Робота зі статичними бібліотеками в C++
    b. Використання маніпуляторів виводу (hex, showbase)
    c. Обробка винятків та помилок
    d. Робота з Git та Code::Blocks

Відповіді на контрольні питання

1. **Мета процесів проектування ПЗ згідно ISO/IEC 12207**
   Стандарт визначає процеси життєвого циклу ПЗ для забезпечення якості, надійності та керованості розробки. Основні задачі: визначення вимог, проектування архітектури, тестування та супровід.
2. **Різниця між функцією та модулем:**
   Функція - це окремий блок коду, що виконує конкретне завдання.
   Модуль - це група пов'язаних функцій та даних, організованих у окремий файл для кращої структуризації коду.
3. **Відмінність функції main()**
   Це обов'язкова точка входу в програму, з якої починається виконання. Інші функції викликаються явно, тоді як main() викликається автоматично при запуску.
4. **Призначення маніпуляторів.** Маніпулятори - керують форматуванням виводу.
5. **Призначення заголовкових файлів.**
   Вони містять оголошення функцій, класів та констант, що дозволяє розділяти код між різними модулями. Використовуються з #include.
6. **Використання iostream.**
   Використано для введення, виведення та керування потоком.
7. **Стандартний простір імен std**
   Містить стандартні об'єкти C++ (cout, cin, string).

8. **Заборонені символи в ідентифікаторах**
   Не можна використовувати: пробіли, спецсимволи (@, #, %), починати з цифри, ключові слова (int, return тощо).
9. **Виведення у шістнадцятковій системі**
   cout << hex << showbase << 255;
10. **Підключення власної бібліотеки**
    Потрібен заголовковий файл с оголошенням, файл реалізації, вказати шляхи в налаштуваннях проекту.
11. **Різниця заголовкового та об'єктного файлів**
    .h містить оголошення, .o/.obj - скомпільований машинний код.
12. **Виведення спецсимволів**
    cout << "√2 ≈ 1.41" (можна навіть скопіювати з юнікоду)
13. **Символьний vs рядковий літерал**
    Символьний: 'A' (1 байт)
    Рядковий: "ABC" (масив символів з \0 в кінці)
14. **Синтаксис прототипу функції**
    float calculate() в .h, float calculate(){return 0} в .cpp
15. **Оператор return**
    Повертає значення з функції
16. **Призначення тестових драйверів**
    Це спеціальні програми для автоматизованого тестування окремих модулів або функцій.
17. **Тестування модулів vs exe-файлів**
    Модульне тестування перевіряє окремі компоненти, тоді як тестування exe - інтегровану роботу всієї програми.

Висновок

У ході виконання лабораторної роботи №7 було набуте практичне застосування алгоритмів обробки масивів даних та символьної інформації мовою програмування C (ISO/IEC 9899:2018).

У першому завданні була реалізована програма для пошуку слова "комп'ютер" у введеному користувачем реченні без урахування регістру символів. Було застосовано функції роботи з рядками, зокрема strstr(), а також переведення символів у нижній регістр для забезпечення коректного пошуку.

У другому завданні реалізована програма для підрахунку кількості заданих натуральних чисел у масиві за допомогою оператора switch. Це дозволило ефективно виконати перевірку на рівність кількох фіксованих значень.

Отримані результати підтвердили правильність роботи програм, а також дозволили закріпити навички роботи з масивами, рядками та умовними операторами.

Відповіді на контрольні питання

1. git init — Ініціалізація нового локального репозиторію в поточній папці. Створює приховану папку .git, яка містить усі дані для відстеження змін.
2. git add (git add .) — Додає файли до індексу (області підготовки) для наступного коміту. git add . додає всі файли в поточній директорії та її піддиректоріях.
3. git commit (git commit -m "текст_коміту") — Фіксує зміни в репозиторії з описом змін. Опція -m дозволяє додати повідомлення про коміт без відкриття текстового редактора.
4. git remote add (git remote add origin) — Додає віддалений репозиторій із псевдонімом origin, щоб можна було надсилати туди зміни.
5. git push (git push) — Надсилає закомічені зміни з локального репозиторію на віддалений репозиторій, прив'язаний до гілки.