Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 9

з навчальної дисципліни

“Базові методології та технології програмування”

РЕАЛІЗАЦІЯ ПРОГРАМНИХ МОДУЛІВ РОЗГАЛУЖЕНИХ

ТА ІТЕРАЦІЙНИХ 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

Габур.А.О

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Коваленко А. С.

Кропивницький – 2025

Мета роботи полягає у набутті грунтовних вмінь і практичних навичок реалізацій технологій модульного програмування, застосування операторів С/С++ арифметичних, логічних, побітових операцій, умови, циклів та вибору під час розроблення статичних бібліотек, заголовкових файлів та програмних засобів у кросплатформовому середовищі Code::Blocks.

ЗАВДАННЯ ДО ЛАБОРАТОРНОЇ РОБОТИ:

1. Реалізувати функції розв’язування задач 9.1–9.3 як складових статичної бібліотеки libModulesПрізвище.а (проект ModulesПрізвище, створений під час виконання лабораторної роботи №8). 2. Реалізувати програмне забезпечення розв’язування задачі 9.4 на основі функцій статичної бібліотеки libModulesПрізвище.а.

ВАРІАНТ 17

Задача 9.1:

Вхід: бал сили вітру за шкалою Бафорта.

Вихід: швидкість та характеристика вітру.

|  |  |  |
| --- | --- | --- |
| БАЛ БАФОРТА | ШВИТКІСТЬ ВІТРУ,м/сек | ХАРАКТЕРИСТИКА |
| 0 | <0,3 | Штиль |
| 1 | 0.3-1.5 | Тихий |
| 2 | 1.6-9.4 | Легкий |
| 3 | 9.4-5.4 | Слабкий |
| 4 | 5.5-7.9 | Помірний |
| 5 | 8.0-10.7 | Свіжий |
| 6 | 10.8-13.8 | Сильний |
| 7 | 13.9-17.1 | Міцний |
| 8 | 17.2-20.7 | Дуже міцний |
| 9 | 20.8-24.4 | Шторм |
| 10 | 24.5-28.4 | Сильний шторм |
| 11 | 28.5-32.6 | Шквальний шторм |
| 12 | =>32.7 | Ураган(буревій) |

Лістинг задачі 9.1:

#include <iostream>

#include <windows.h>

#include <string>

using namespace std;

int main() {

    SetConsoleOutputCP(65001);

    int бал;

    while (true) {

        cout << "\nВведіть бал сили вітру за шкалою Бофорта (0–12): ";

        cin >> бал;

        if (бал < 0 || бал > 12) {

            cout << "Некоректне значення бала. Введіть число від 0 до 12." << endl;

            continue;

        }

        string швидкість, характеристика;

        switch (бал) {

            case 0: швидкість = "< 0.3"; характеристика = "Штиль"; break;

            case 1: швидкість = "0.3 – 1.5"; характеристика = "Тихий вітер"; break;

            case 2: швидкість = "1.6 – 3.3"; характеристика = "Легкий вітер"; break;

            case 3: швидкість = "3.4 – 5.4"; характеристика = "Слабкий вітер"; break;

            case 4: швидкість = "5.5 – 7.9"; характеристика = "Помірний вітер"; break;

            case 5: швидкість = "8.0 – 10.7"; характеристика = "Свіжий вітер"; break;

            case 6: швидкість = "10.8 – 13.8"; характеристика = "Сильний вітер"; break;

            case 7: швидкість = "13.9 – 17.1"; характеристика = "Міцний вітер"; break;

            case 8: швидкість = "17.2 – 20.7"; характеристика = "Дуже міцний вітер"; break;

            case 9: швидкість = "20.8 – 24.4"; характеристика = "Шторм"; break;

            case 10: швидкість = "24.5 – 28.4"; характеристика = "Сильний шторм"; break;

            case 11: швидкість = "28.5 – 32.6"; характеристика = "Шквальний шторм"; break;

            case 12: швидкість = "≥ 32.7"; характеристика = "Ураган (буревій)"; break;

        }

        cout << "Швидкість вітру: " << швидкість << " м/с" << endl;

        cout << "Характеристика: " << характеристика << endl;

        char продовжити;

        cout << "\nБажаєте продовжити (y/n)? ";

        cin >> продовжити;

        if (продовжити == 'n' || продовжити == 'N') {

            cout << "Програма завершена." << endl;

            break;

        }

    }

    return 0;

}

Задача 9.2:

Вхід: розмір чоловічих шкарпеток за українською системою

Вихід: відповідники розмірів шкарпеток у системах США і ЄС

Таблиця відповідностірозмірів шкарпеток(чол.)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ЄС | 37/38 | 39/40 | 41/42 | 43/44 | 45/46 |
| Великобританія, США | 8 | 9 | 10 | 11 | 12 |
| Україна | 23 | 25 | 27 | 29 | 31 |

Лістинг задачі 9.2:

#include <iostream>

#include <windows.h>

using namespace std;

int main() {

    SetConsoleOutputCP(65001);

    int rozmirUA;

    cout << "Введіть розмір чоловічих шкарпеток за українською системою (23, 25, 27, 29, 31): ";

    cin >> rozmirUA;

    string rozmirEU, rozmirUS;

    switch (rozmirUA) {

        case 23:

            rozmirEU = "37/38";

            rozmirUS = "8";

            break;

        case 25:

            rozmirEU = "39/40";

            rozmirUS = "9";

            break;

        case 27:

            rozmirEU = "41/42";

            rozmirUS = "10";

            break;

        case 29:

            rozmirEU = "43/44";

            rozmirUS = "11";

            break;

        case 31:

            rozmirEU = "45/46";

            rozmirUS = "12";

            break;

        default:

            cout << "Помилка: невідомий розмір. Будь ласка, введіть один із таких: 23, 25, 27, 29, 31." << endl;

            return 1;

    }

    cout << "Розмір у системі ЄС: " << rozmirEU << endl;

    cout << "Розмір у системі США/Великобританії: " << rozmirUS << endl;

    return 0;

}

Задача 9.3:

Вхід: натуральне число N від 0 до 7483650.

Вихід: якщо біт D7 числа N рівний 1,клькість двійкових нулів у ньому,

інакше – суму двійкових одиниць\*.

\*під час підрахунку кількості бінарних 0 або 1 рекомендовано використати тернарний оператор «? :»

Лістинг до задачі 9.3:

#include <iostream>

#include <bitset>

#include <windows.h>

using namespace std;

int main() {

    SetConsoleOutputCP(65001);

    unsigned int N;

    cout << "Введіть натуральне число від 0 до 7483650: ";

    cin >> N;

    if (N > 7483650) {

        cout << "Помилка: число перевищує 7483650!" << endl;

        return 1;

    }

    bitset<32> bits(N);

    bool D7 = bits[7];

    int count0 = 0, sum1 = 0;

    for (int i = 0; i < 32; ++i) {

        if (bits[i] == 0)

            count0++;

        else

            sum1++;

    }

    // Тернарний оператор:

    D7 ? cout << "Кількість двійкових нулів: " << count0 << endl

       : cout << "Сума двійкових одиниць: " << sum1 << endl;

    return 0;

}

Задача 9.4:

За введеним користувачем символом “u” викликається s\_calculation(),  
“o” — функція задачі 9.1, “p” — функція задачі 9.2, “a” — функція задачі 9.3;  
якщо користувач вводить інші символи, вони ігноруються, при чому  
видається звуковий сигнал про помилкове введення. Після цього, якщо  
користувач за запитом додатка вводить символ “s” або “S”, відбувається  
вихід з програми, інакше — виконання програми повторюється.

Лістинг задачі 9.4:

#include <iostream>

#include <windows.h>

#include <limits>

#include "Modules/Gabur.h"

#include "Modules/Gabur.cpp"

#include "9\_1.cpp"

#include "9\_2.cpp"

#include "9\_3.cpp"

using namespace std;

void task9\_4() {

    double x, y, z;

    char a, b;

    cout << "\nВведіть значення x, y, z (через пробіл): ";

    cin >> x >> y >> z;

    cout << "Введіть два логічні символи (наприклад: 1 0): ";

    cin >> a >> b;

    printAuthor();

    printValues(x, y, z);

    double s = calculateS(x, y, z);

    cout << "Результат обчислення S: " << s << endl;

    printLogicalResult(a, b);

}

int main() {

    SetConsoleOutputCP(65001); // українська мова в консолі

    char команднийСимвол;

    while (true) {

        cout << "\nВведіть команду (u-завдання 4,o-бал бофорда,p-розмір носка,a-двійкові числа): ";

        cin >> команднийСимвол;

        cin.ignore(numeric\_limits<streamsize>::max(), '\n'); // 🔧 очищаємо буфер

        switch (команднийСимвол) {

            case 'u': task9\_4(); break;

            case 'o': task1(); break;

            case 'p': task2(); break;

            case 'a': task3(); break;

            default:

                cout << "\aНевідома команда! (ігноровано)" << endl;

                continue;

        }

        char вихід;

        cout <<"\nДля виходу введіть 's' або 'S',або";

        cout << "\nВведіть команду (u-завдання 4,o-бал бофорда,p-розмір носка,a-двійкові числа): ";

        cin >> вихід;

        cin.ignore(numeric\_limits<streamsize>::max(), '\n'); // 🔧 знову очищаємо буфер

        if (вихід == 's' || вихід == 'S') {

            cout << "Завершення програми." << endl;

            break;

        }

    }

    return 0;

}

Лістинг до тест драйвера:

#include <iostream>

#include <string>

#include <bitset>

#include <cmath>

#include <iomanip>

#include <windows.h>

#include <cstdlib>

using namespace std;

// task1 — опис вітру за шкалою Бофорта

string getWindDescription(int бал) {

    if (бал < 0 || бал > 12) return "Помилка";

    switch (бал) {

        case 0: return "Штиль";

        case 1: return "Тихий вітер";

        case 2: return "Легкий вітер";

        case 3: return "Слабкий вітер";

        case 4: return "Помірний вітер";

        case 5: return "Свіжий вітер";

        case 6: return "Сильний вітер";

        case 7: return "Міцний вітер";

        case 8: return "Дуже міцний вітер";

        case 9: return "Шторм";

        case 10: return "Сильний шторм";

        case 11: return "Шквальний шторм";

        case 12: return "Ураган (буревій)";

    }

    return "Помилка";

}

// task2 — розмір шкарпеток

string getSocksSize(int rozmirUA) {

    switch (rozmirUA) {

        case 23: return "37/38";

        case 25: return "39/40";

        case 27: return "41/42";

        case 29: return "43/44";

        case 31: return "45/46";

        default: return "Помилка";

    }

}

// task3 — бітовий аналіз

string binaryAnalysis(unsigned int N) {

    if (N > 7483650) return "Помилка";

    bitset<32> bits(N);

    bool D7 = bits[7];

    int count0 = 0, count1 = 0;

    for (int i = 0; i < 32; ++i) {

        if (bits[i] == 0) count0++;

        else count1++;

    }

    return D7 ? "count0=" + to\_string(count0) : "sum1=" + to\_string(count1);

}

// task4 — математичне обчислення

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

    if ((x - y) <= 0 || y == 0) return NAN;

    double numerator = 3.14159265 \* pow(x, 2);

    double denominator = x + z / (2 \* pow(y, 2));

    if (denominator == 0) return NAN;

    return log(x - y) + sqrt(numerator / denominator);

}

// Автор

void printAuthor() {

    cout << "© Габур Артем" << endl;

}

// тест для string

// тест для string

void runTest(const string& name, const string& actual, const string& expected) {

    cout << name << ": ";

    if (actual == expected)

        cout << "✅ (Очікувалось: \"" << expected << "\", отримано: \"" << actual << "\")" << endl;

    else

        cout << "❌ (Очікувалось: \"" << expected << "\", отримано: \"" << actual << "\")" << endl;

}

// тест для double

void runTestDouble(const string& name, double actual, double expected, double epsilon = 1e-6) {

    cout << name << ": ";

    if (isnan(expected) && isnan(actual)) {

        cout << "✅ (Очікувалось: NAN, отримано: NAN)" << endl;

    }

    else if (fabs(actual - expected) < epsilon) {

        cout << "✅ (Очікувалось: " << expected << ", отримано: " << actual << ")" << endl;

    }

    else {

        cout << "❌ (Очікувалось: " << expected << ", отримано: " << actual << ")" << endl;

    }

}

int main() {

    SetConsoleOutputCP(65001);

    cout << "\n--- ТЕСТИ ЗАДАЧІ 9.1 ---\n" << endl;

    // task1

    runTest("Task1 Бал 0", getWindDescription(0), "Штиль");

    runTest("Task1 Бал 3", getWindDescription(3), "Слабкий вітер");

    runTest("Task1 Бал 12", getWindDescription(12), "Ураган (буревій)");

    runTest("Task1 Бал -1", getWindDescription(-1), "Помилка");

    cout << "\n--- ТЕСТИ ЗАДАЧІ 9.2 ---\n" << endl;

    runTest("UA=23", getSocksSize(23), "37/38");

    runTest("UA=25", getSocksSize(25), "39/40");

    runTest("UA=27", getSocksSize(27), "41/42");

    runTest("UA=29", getSocksSize(29), "43/44");

    runTest("UA=31", getSocksSize(31), "45/46");

    runTest("UA=22 (некоректне)", getSocksSize(22), "Помилка");

    cout << "\n--- ТЕСТИ ЗАДАЧІ 9.3 ---\n" << endl;

    // task3

    runTest("Task3 N = 15", binaryAnalysis(15), "sum1=4");

    runTest("Task3 N = 256", binaryAnalysis(256), "sum1=1");

    runTest("Task3 N = 8000000", binaryAnalysis(8000000), "Помилка");

    cout << "\n--- ТЕСТИ ЗАДАЧІ 9.4 ---\n" << endl;

    // task4

    double x1 = 5.0, y1 = 2.0, z1 = 3.0;

    double expected1 = log(x1 - y1) + sqrt((3.14159265 \* pow(x1, 2)) / (x1 + z1 / (2 \* pow(y1, 2))));

    runTestDouble("Task4 x=5, y=2, z=3", calculateS(x1, y1, z1), expected1);

    double x2 = 4.0, y2 = 4.0, z2 = 2.0;

    runTestDouble("Task4 x=4, y=4, z=2", calculateS(x2, y2, z2), NAN);

    // автор

    printAuthor();

    system("pause");

    return 0;

}

Відповіді на контрольні питання:

1. Оператором if...else можна повноцінно замінити тернарний оператор ? :.

// Тернарний оператор:

int x = (a > b) ? a : b;

// Заміна на if...else:

int x;

if (a > b)

x = a;

else

x = b;

if...else є більш гнучким і дозволяє виконувати складніші дії, але тернарний оператор зручніший для простих виразів.

2.Пріоритет операцій — це порядок, у якому виконуються операції без дужок. Наприклад, \* має вищий пріоритет за +.

Асоціативність — визначає напрям виконання однакових за пріоритетом операцій (зліва направо чи навпаки). Наприклад, a - b - c обчислюється як (a - b) - c.

3. Змінні, оголошені в тілі циклу або умови, мають локальну область видимості, тобто існують тільки всередині блоку {}.

if (true) {

int x = 5;

}

// x тут недоступна — помилка компіляції

4.Арифметичні (+, -, \*, /, %): лівоасоціативні

Логічні (&&, ||): лівоасоціативні

Порозрядні (&, |, ^, <<, >>): лівоасоціативні

Інкремент/декремент (++, --): префікс — правоасоціативні, постфікс — вище в пріоритеті

Тернарний ? :: правоасоціативний

Операції порівняння (<, >, ==, !=): лівоасоціативні

5. Тернарний оператор доцільний для простих умовних присвоєнь або повернень значень:

int max = (a > b) ? a : b;

Використовується для стислості коду, якщо немає потреби в кількох командах.

6. cnt--; зменшує значення змінної на 1. Якщо cnt = 5, то після цієї інструкції cnt == 4.

7. const-змінна — це змінна, значення якої не можна змінити після ініціалізації.  
Правило: оголошуйте змінну const, якщо її значення не повинне змінюватися у програмі:

const int DAYS\_IN\_WEEK = 7;

8. Операнди логічних операторів можуть бути будь-якого типу, але інтерпретуються як булеві. Наприклад, 0 — false, усе інше — true.

9. bool cnt = !!0;  
Оператор ! перетворює 0 у true, тобто !0 == 1, ще один ! — !!0 == 0.  
Отже, cnt == false.

10. У циклі for ініціалізація записується у першій частині:

for (int i = 0; i < 10; ++i) { ... }

Правило: ініціалізація виконується один раз перед початком циклу.

11.for (char c; std::cin >> c && c != '!'; ) {

// дія

}

12.Повне розгалуження: if (умова) { ... } else { ... }

Неповне: if (умова) { ... }  
Недопустимі операції: ті, що не повертають логічне або числове значення. Наприклад, заборонено використання void-функції у виразі.

13.Логічні операції: AND (&&), OR (||), NOT (!)

Операнди: будь-який тип, але інтерпретуються як логічні

Результат: тип bool (0 або 1)

14. ++cnt -= 1;  
Це некоректний вираз — компілятор видасть помилку, бо ++cnt повертає тимчасове значення, до якого не можна застосувати -=.

15.const double PI = 3.14159;

У модулі (файлі .cpp або .h) така константа часто оголошується з constexpr або static.

16. break — оператор, що перериває виконання циклу або switch-блоку.  
Особливість: негайний вихід з конструкції, де він використовується.

17.Унарний: ~ (NOT)

Бінарні: &, |, ^, <<, >>

Типи: цілі типи (int, char, long)

Приклад:

int a = 5 & 3; // a = 1

int b = ~5; // b = -6 (інверсія біта)

18.for — відоме число ітерацій

while — умова перевіряється перед входом, число ітерацій заздалегідь невідоме

do...while — хоча б одна ітерація гарантована

19. continue варто використовувати, коли потрібно пропустити поточну ітерацію й перейти до наступної:

for (int i = 0; i < 10; ++i) {

if (i % 2 == 0) continue;

std::cout << i << " ";

}

20. У Code::Blocks для створення статичних бібліотек:

1. Створити новий проект типу *Static Library*
2. Додати .cpp і .h файли з функціями
3. Зібрати проект (отримаєте .a файл)
4. Підключити .a файл до основного проекту через налаштування "Linker settings"

**Аргументи досягнення мети лабораторної роботи:**

1. Використання функцій для модульної структури коду.
2. Застосування циклів для обробки даних.
3. Реалізація умовних операторів для аналізу введення.
4. Робота з масивами для зберігання даних.
5. Підключення бібліотек для розширення можливостей.
6. Обробка вводу та виводу для взаємодії з користувачем.
7. Використання математичних операцій для обчислень.
8. Застосування побітових операцій для маніпуляцій.
9. Використання тернарного оператора для компактності.
10. Створення меню для зручного вибору функцій.
11. Форматування виводу через escape-послідовності.
12. Налаштування кодування за допомогою системних викликів.
13. Використання констант для фіксованих значень.
14. Обробка помилок для стабільності програми.
15. Повторення дій через цикли до виконання умови.
16. Передача даних у функції через параметри.
17. Повернення результатів із функцій.
18. Робота з різними типами даних для обчислень.
19. Зберігання послідовностей у масивах.
20. Ітерація по масивах за допомогою циклів.
21. Перевірка умов через умовні оператори.
22. Використання математичних функцій для розрахунків.
23. Маніпуляція бітами через побітові операції.
24. Вибір варіантів за допомогою тернарного оператора.
25. Реалізація меню через оператор switch.
26. Використання звукових сигналів у виводі.
27. Налаштування середовища системними командами.
28. Зберігання констант для повторного використання.
29. Захист від некоректного вводу користувача.
30. Виконання повторюваних дій у циклах.
31. Передача аргументів у функції.
32. Отримання результатів через повернення значень.
33. Обчислення з різними типами даних.
34. Організація даних у масивах.
35. Обробка масивів через ітерації.
36. Аналіз умов для логічних рішень.
37. Використання вбудованих математичних функцій.
38. Робота з бітами для оптимізації.
39. Спрощення коду тернарним оператором.
40. Створення меню з вибором опцій.
41. Форматування тексту у виводі.
42. Налаштування програми системними викликами.
43. Використання констант для чіткості коду.
44. Виявлення та обробка помилок вводу.
45. Автоматизація задач через цикли.
46. Передача параметрів для гнучкості функцій.
47. Отримання результатів із функцій для обробки.
48. Використання типів даних для точних обчислень.
49. Зберігання наборів даних у масивах.
50. Обробка елементів масиву циклами.