МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЦЕНТРАЛЬНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ

ЛАБОРАТОРНА РОБОТА №9

З ДИСЦИПЛІНИ

БАЗОВІ МЕТОДОЛОГІЇ ТА ТП

ТЕМА ЛАБОРАТОРНОЇ РОБОТИ:

**РЕАЛІЗАЦІЯ ПРОГРАМНИХ МОДУЛІВ РОЗГАЛУЖЕНИХ ТА ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ**

Студент:Новак П.А.

Група КІ-24

Викладач: Коваленко А.С.

м.Кропивницький

2025

**Мета роботи:** полягає у набутті ґрунтовних вмінь і практичних навичок реалізації технології 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.

**ОБЛАДНАННЯ МАТЕРІАЛИ, ПРОГРАМНІ ЗАСОБИ**

Для виконання лабораторної роботи необхідні:

– персональний комп’ютер з ОС Windows XP / Vista / 7 / 8.x / 10, Linux 32-bit / 64-bit або Mac OS X;

– вільне кросплатформове середовище розроблення програмного забезпечення Code::Blocks (www.codeblocks.org) для відповідної платформи: Windows XP / Vista / 7 / 8.x / 10, Linux 32-bit / 64-bit, або Mac OS X;

– текстовий редактор (OpenOffice Writer, Microsoft Word або ін.);

– файл-шаблон тестового набору Artifact\_TEST\_SUITE\_lab.doc;

– Git-репозиторій <https://github.com/odorenskyi/student-name>.

**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

**ЗАВДАННЯ 9.1-9.3**

**ModulesNovak.cpp -** реалізація функцій

#include "ModulesNovak.h"

#include <cmath>

#include <bitset>

#include <map>

#include <iostream>

#include "Windows.h"

double calculate\_salary(int days) {

double gross = days \* 8 \* 450;

double tax = gross \* 0.15;

double pension = gross \* 0.02;

double unemployment = gross \* 0.006;

double insurance = gross \* 0.01;

double net = gross - (tax + pension + unemployment + insurance);

return net;

}

std::string convert\_size(int ua\_size) {

std::map<int, std::string> size\_map = {

{23, "37/38"}, {25, "39/40"}, {27, "41/42"}, {29, "43/44"},

{31, "45/46"}

};

return size\_map.count(ua\_size) ? size\_map[ua\_size] : "Unknown";

}

std::pair<int, int> count\_binary\_digits(int N) {

std::bitset<32> b(N);

int zeros = 0, ones = 0;

for (int i = 0; i < 32; ++i)

(b[i] ? ones : zeros)++;

return { zeros, ones };

}

void s\_calculation(char option) {

if (option == 'v') {

int days;

std::cout << "Введіть кількість відпрацьованих днів: ";

std::cin >> days;

std::cout << "Зароблені гроші: " << calculate\_salary(days) << " UAH\n";

}

else if (option == 'm') {

int size;

std::cout << "Введіть UA розмір ноги: ";

std::cin >> size;

std::cout << "Розмір в US/UK/EU: " << convert\_size(size) << "\n";

}

else if (option == 'q') {

int N;

std::cout << "Введіть число: ";

std::cin >> N;

std::pair<int, int> result = count\_binary\_digits(N);

std::cout << "Двійкові нулі: " << result.first << ", ones: " << result.second << "\n";

}

else {

std::cerr << "Unknown command.\n";

}

}

**ModulesNovak.h** – заголовковий файл

#ifndef MODULES\_PRIZVISHE\_H

#define MODULES\_PRIZVISHE\_H

#include <string>

#include <utility>

double calculate\_salary(int days);

std::string convert\_size(int ua\_size);

std::pair<int, int> count\_binary\_digits(int N);

void s\_calculation(char option);

#endif

**Задача 9.1**

**Умова:**  
Згідно з трудовим договором, кожен працівник отримує заробітну плату відповідно до кількості відпрацьованих днів у звітному періоді. За один відпрацьований день працівник отримує фіксовану оплату 450 грн за 8-місячний період роботи. Визначити:

* кількість відпрацьованих днів за звітний період,
* суму зарплати, яку має отримати працівник, враховуючи податки та внески.

**Постановка задачі:**  
Розробити програму, яка за вхідними даними (кількістю відпрацьованих днів, ставками податків та внесків) обчислює зарплату працівника з урахуванням усіх податків і відрахувань.

**Задача 9.2**

**Умова:**  
Надається таблиця відповідності розмірів чоловічих шкарпеток за українською системою та системами США і ЄС. Вхідні дані — розмір жіночих шкарпеток за українською системою. Вивести відповідний розмір шкарпеток у системах США та ЄС.

**Постановка задачі:**  
Написати програму, яка, маючи розмір жіночих шкарпеток у українській системі (число від 23 до 31), визначить і виведе відповідний розмір шкарпеток у системах США та ЄС, використовуючи таблицю співвідношень.

**Задача 9.3**

**Умова:**  
Дано натуральне число N у діапазоні від 0 до 7483650. Необхідно визначити:

* кількість цифрових одиниць у двійковому представленні числа N (кількість цифр "1").

**Постановка задачі:**  
Розробити програму, яка для заданого числа N обчислює кількість двійкових одиниць у його бінарному представленні.

**TestDriver -** модульне тестування

#include <iostream>

#include <locale>

#include <windows.h>

#include "ModulesNovak.h"

void run\_test(wchar\_t option, const std::wstring& description) {

std::wcout << L"\n[Тест] " << description << L" (опція '" << option << L"')\n";

s\_calculation(static\_cast<char>(option));

}

int main() {

SetConsoleOutputCP(CP\_UTF8);

SetConsoleCP(CP\_UTF8);

std::locale::global(std::locale("uk\_UA.UTF-8"));

std::wcout << L"=== Тест-драйвер ===\n";

run\_test(L'v', L"Розрахунок зарплати");

run\_test(L'm', L"Конвертація розміру");

run\_test(L'q', L"Підрахунок бітів");

run\_test(L'x', L"Невідома команда");

return 0;

}

**ЗАВДАННЯ 9.4**

**Novak\_Task.cpp –** програмна реалізація

#include <iostream>

#include <windows.h>

#include "ModulesNovak.h"

int main() {

SetConsoleOutputCP(CP\_UTF8);

SetConsoleCP(CP\_UTF8);

std::locale::global(std::locale("uk\_UA.UTF-8"));

wchar\_t choice;

std::wcout << L"Програма задачі 9.4\n";

do {

std::wcout << L"\nМеню:\n";

std::wcout << L"v – обчислення зарплати\n";

std::wcout << L"m – конвертація розміру\n";

std::wcout << L"q – підрахунок бітів\n";

std::wcout << L"x – вихід\n";

std::wcout << L"Ваш вибір: ";

std::wcin >> choice;

if (choice != L'x') {

s\_calculation(choice);

}

} while (choice != L'x');

return 0;

}

**Задача 9.4**

**Умова:**  
За введеним користувачем символом “v”, викликається функція s\_calculation(). Параметри функції:

* n — номер задачі (від 9.1 до 9.3),
* m — номер задачі 9.2,
* q — номер задачі 9.3.  
  Якщо користувач вводить інші символи, програма повинна видавати звуковий сигнал про помилкове введення.

**Постановка задачі:**  
Розробити основний програмний цикл, який за введеним символом виконує відповідну функцію або реагує сигналом про неправильний ввід, у разі некоректного символу.

**АРГУМЕНТИ**

1. Розвиває ґрунтовні навички програмування.
2. Навчає реалізовувати модульні програми.
3. Формує практичні навички роботи з операторами.
4. Вчить застосовувати арифметичні оператори.
5. Вчить користуватись логічними операторами.
6. Практикує побітові операції.
7. Розвиває вміння писати умови та цикли.
8. Формує навички створення статичних бібліотек.
9. Вчить працювати із заголовковими файлами.
10. Опановує засоби розробки в Code::Blocks.
11. Застосовує кросплатформову розробку.
12. Підвищує рівень кросплатформеної компетенції.
13. Розвиває навички застосування операторів у реальних задачах.
14. Вчить створювати багатофункціональні програми.
15. Формує поняття структурованого програмування.
16. Надає досвід з підключенням та використанням бібліотек.
17. Розширює технічне мислення.
18. Забезпечує розвиток творчих і технічних навичок.
19. Готує до роботи в реальних проектних умовах.
20. Підвищує компетенцію з обґрунтування вибору інструкцій.
21. Вчить модульності у програмуванні.
22. Розвиває навички системного аналізу задач.
23. Застосовує теоретичні знання на практиці.
24. Вчить працювати з стандартними бібліотеками.
25. Зміцнює навички роботи з інструментами розробки.
26. Підвищує рівень професійної підготовки.
27. Формує навички документування коду.
28. Вчить створювати високорівневі структуровані програми.
29. Розвиває алгоритмічне мислення.
30. Надає досвід оптимізації програм.
31. Формує навички роботи з системними викликами.
32. Вчить проводити тестування модулів.
33. Підвищує розуміння роботи комп’ютера.
34. Забезпечує практичне засвоєння теоретичних знань.
35. Вчить написанню ефективного та безпечного коду.
36. Розвиває навички роботи з репозиторіями і версіями.
37. Забезпечує підготовку до серйозних проектів.
38. Вчить працювати у команді.
39. Розвиває відповідальність за якість коду.
40. Формує навички розв’язання складних задач.
41. Вчить спільній роботі із файлами та даними.
42. Отримує досвід у створенні системних утиліт.
43. Підвищує рівень технічної грамотності.
44. Вчить використовувати функції для структурування коду.
45. Формує розуміння архітектури програм.
46. Розвиває системне та логічне мислення.
47. Підготує до подальшого вивчення мов низького рівня.
48. Вчить писати переносний і масштабовний код.
49. Формує вміння застосовувати знання для автоматизації задач.
50. Розширює можливості створення власних бібліотек і модулів.

**КОНТРОЛЬНІ ЗАПИТАННЯ**

**1. Яким оператором С/С++ можливо повноцінно замінити тернарний оператор?**  
Поверненню функціональності тернарного оператора може служити конструкція «якщо — інакше» (if-else), яка дає можливість виконати тотожне логічне рішення. Це повна форма умовної інструкції, яка здатна замінити тернарник у будь-якій ситуації.

**2. Що в програмуванні розуміють під пріоритетом виконання операцій та асоціативністю?**  
Пріоритет визначає порядок обробки операторів у виразі без дужок — тобто, які операції будуть виконані першими. Асоціативність — це правило послідовності виконання операторів однієї пріоритетної групи (зліва направо або справа наліво), якщо їх декілька.

**3. Яку область видимості мають змінні, оголошені в тілі циклу або умови?**  
Змінні, оголошені в межах циклу або в тілі умовної конструкції, мають локальну область видимості. Це означає, що вони видимі і можуть використовуватись тільки у межах цього блоку — за його межами вони недоступні і перестають існувати після завершення цього блоку.

**4. Якою є асоціативність основних операторів у мовах C/ C++?**  
Асоціативність більшості арифметичних, логічних і побітових операторів — лінійна і здебільшого зліва направо. Інкремент і декремент мають особливу увагу: префіксні форми мають праву асоціативність, а постфіксні — леву. Тернарний оператор виконується справа наліво. Оператори порівняння — зліва направо.

**5. Перелічіть ситуації, коли доцільно використовувати тернарний оператор, і наведіть приклад його запису.**  
Тернарний оператор зручно застосовувати для швидкого вибору між двома значеннями або діями на основі однієї умови — і тоді можна уникнути довгих конструкцій «if-else». Наприклад, для вибору максимального з двох чисел, або для присвоєння значення залежно від умови.

**6. Яке значення міститиме змінна cnt після виконання cnt--;?**  
Змінна cnt буде зменшена на один. Якщо вона була, наприклад, 5, то після виконання цієї інструкції вона стане 4.

**7. Чим константна змінна, оголошена за допомогою const, відрізняється від звичайної?**  
Константна змінна — це така, що після ініціалізації її значення не може бути змінене в процесі виконання програми. Її оголошують, щоб запобігти випадковим або навмисним змінам значення, що підвищує безпеку та стабільність коду.

**8. Яких типів можуть бути операнди логічних операторів?**  
Як правило, операнди можуть бути цілочисельними, булевими типами або будь-якими іншими типами, які можна перетворити у булеве значення (істина або хиба). В більшості випадків використовуються цілі числа і булевий тип.

**9. Яке значення міститиме cnt після bool cnt = !!0;?**  
Значення буде хибним (false), тому що подвійне заперечення булевих значень !! зводить будь-яке число до логічного істина або хиби. Для нуля — хибно.

**10. Сформулюйте правило запису виразу ініціалізації у циклах for у C++.**  
У циклі for перша частина підпису (ініціалізація) має містити вираз, що задає початкове значення змінної або змінних, які використовуються в циклі,

**11. Наведіть приклад запису циклу for, який виконується до введення користувачем символу «!» без використання жодної змінної, оголошеної до цього циклу.**  
Можна використовувати нескінченний цикл, що буде виконуватися доти, поки користувач не введе символ «!», і при цьому не потрібно оголошувати змінну перед циклом.

**12. Яким є синтаксис запису повного й неповного розгалуження у C++?**  
Повне розгалуження — це структура, яка включає інструкцію «if» і «else». Неповне — це «if» без «else». У логічних виразах можна використовувати будь-які логічні операції, крім тих, що є недопустимими у логічних виразах (наприклад, операція присвоєння у виразах умови недопустима). Для істинних і хибних виразів, що складаються із логічних операторів — допустимі.

**13. Які логічні операції реалізовано в мовах C++?**  
У C++ реалізовані основні логічні оператори: логічне І (AND), логічне ІЛІ (OR) та логічне заперечення (NOT). Вони застосовуються до будь-яких типових операндів, які перетворюються у булеві значення. Результат — булевий тип.

**14. Яке значення буде у змінній cnt після виконання ++cnt -= 1;?**  
Розібравши цей вираз, відзначимо, що значення cnt спершу збільшується на 1, а потім із отриманого результату віднімається 1, що повертає змінну у її початкове значення, тобто cnt залишиться без змін.

**15. Наведіть приклад опису константної змінної у C++.**  
Константна змінна оголошується з використанням ключового слова const і ініціалізується разом із оголошенням. Наприклад, оголошення цілого числового константного значення.

**16. Яке призначення має оператор break у C++ та в чому полягає особливість його виконання?**  
Оператор break використовується для раптового виходу з циклу або з оператора switch. Особливість його полягає у тому, що він негайно припиняє виконання поточного циклу або switch і передає управління на перший після нього інструкції.

**17. Які логічні порозрядні операції реалізовано у C++?**  
У C++ реалізовані унарні та бінарні порозрядні оператори: побітове І (&), побітове ІЛІ (|), виключне або (^), зсув вліво (<<) і зсув вправо (>>). Вони застосовуються до цілочисельних типів і працюють з окремими бітами.

**18. Сформулюйте критерії вибору конкретного виду циклу (for, while, do…while) для опису повторення.**  
Обирати цикл слід залежно від специфіки задачі:

* for — підходить для заздалегідь відомої кількості повторів.
* while — коли кількість повторів невідома і потрібно повторювати доти, поки виконується умова.
* do while — коли потрібно виконати тіло циклу принаймні один раз, і далі повторювати при виконанні умови.

**19. Випадки і ситуації, коли слід застосовувати оператор continue у алгоритмах.**  
continue використовують для пропуску поточного кроку циклу і переходу до наступної ітерації, наприклад — щоб ігнорувати непотрібні або некоректні значення у процесі обробки даних.