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

ВИКОНАВ

Студент(КБ-24)

Сумар І.С

ПЕРЕВІРИВ

викладач Коваленко А. С.

Кропивницький – 2025

Мета роботи набуття 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

**Варіант №24**

**Завдання 9.1**

Банк укладає угоди на відкриття депозитних вкладів у гривнях строком на півроку під 16% річних або на рік під 18% річних.

Вхід: сума депозиту та строк дії угоди (кількість місяців).

Вихід: сума щомісячних виплат відсотків від депозиту (грн.) та загальна сума відсотків за весь строк дії договору

**Аналіз і постановка задачі**

**Опис: Банк укладає угоди на відкриття депозитних вкладів у гривнях строком на півроку (6 місяців) під 16% річних або на рік (12 місяців) під 18% річних. Необхідно обчислити щомісячні виплати відсотків та загальну суму відсотків за весь строк дії договору.**

**Постановка задачі:**

**Вхідні дані:**

**Сума депозиту (грн).**

**Строк дії угоди (кількість місяців: 6 або 12).**

**Вихідні дані:**

**Сума щомісячних виплат відсотків від депозиту (грн).**

**Загальна сума відсотків за весь строк дії договору (грн).**

**Умови:**

**Якщо строк дії угоди 6 місяців, річна відсоткова ставка становить 16%.**

**Якщо строк дії угоди 12 місяців, річна відсоткова ставка становить 18%.**

**Алгоритм обчислення:**

**Визначити річну відсоткову ставку залежно від строку (16% або 18%).**

**Обчислити щомісячну відсоткову ставку: річна ставка ÷ 12.**

**Обчислити щомісячну виплату: сума депозиту × щомісячна відсоткова ставка.**

**Обчислити загальну суму відсотків: щомісячна виплата × кількість місяців.**

**Аналіз і постановка задачі 9.2**

**Вхід: середньодобові показники швидкості переміщення повітряних мас (вітру, м/сек) у Кропивницькому за січень.**

**Вихід: найбільша швидкість вітру в балах Бофорта, зафіксована в Кропивницькому протягом січня.**

**План реалізації: написати функцію, яка приймає масив середньодобових показників швидкості вітру за січень, знаходить максимальне значення серед них, а потім визначає відповідний бал Бофорта за допомогою умовних операторів або таблиці відповідності.**

**Аналіз задачі:**

**Необхідно знайти максимальну швидкість вітру за січень.**

**Перевести цю швидкість у бали Бофорта за заданою таблицею.**

**Постановка задачі:**

**Отримати масив середньодобових показників швидкості вітру за січень.**

**Знайти максимальне значення в цьому масиві.**

**Визначити бал Бофорта для цієї швидкості за таблицею:**

**0: <0.3 м/сек**

**1: 0.3-1.5 м/сек**

**2: 1.6-3.3 м/сек**

**3: 3.4-5.4 м/сек**

**4: 5.5-7.9 м/сек**

**5: 8.0-10.7 м/сек**

**6: 10.8-13.8 м/сек**

**7: 13.9-17.1 м/сек**

**8: 17.2-20.7 м/сек**

**9: 20.8-24.4 м/сек**

**10: 24.5-28.4 м/сек**

**11: 28.5-32.6 м/сек**

**12: ≥32.7 м/сек**

**Вивести бал Бофорта.**

**План реалізації:**

**Знайти максимальне значення в масиві.**

**Використати умовні оператори для визначення балу Бофорта.**

**Вивести результат.**

**Аналіз і постановка задачі 9.3**

**Вхід: натуральне число N від 0 до 9008000.**

**Вихід: якщо біт D13 числа N рівний 0, кількість двійкових нулів у ньому, інакше — кількість двійкових одиниць.**

**План реалізації: написати функцію, яка приймає число N, перевіряє значення біта D13, і залежно від цього підраховує кількість 0 або 1 у двійковому представленні числа.**

**Аналіз задачі:**

**Необхідно працювати з двійковим представленням числа.**

**Біт D13 — це 14-й біт зліва (починаючи з 0).**

**Залежно від значення D13, підрахувати кількість 0 або 1.**

**Постановка задачі:**

**Отримати число N.**

**Перевести N у двійкове представлення.**

**Визначити значення біта D13.**

**Якщо D13 == 0, підрахувати кількість 0 у двійковому представленні.**

**Якщо D13 == 1, підрахувати кількість 1 у двійковому представленні.**

**Вивести результат.**

**План реалізації:**

**Перевести число у двійкове представлення.**

**Визначити біт D13.**

**Підрахувати кількість 0 або 1 залежно від D13.**

**Вивести результат.**

**Аналіз і постановка задачі 9.4**

**Вхід: символ, введений користувачем.**

**Вихід: виклик відповідної функції або звуковий сигнал.**

**План реалізації: написати програму, яка зчитує символ від користувача і викликає відповідну функцію або видає звуковий сигнал.**

**Аналіз задачі:**

**Програма має реагувати на введений символ.**

**Залежно від символу, викликати різні функції.**

**Якщо символ не відповідає жодній функції, видати звуковий сигнал.**

**Після виконання, запитати користувача про продовження або завершення.**

**Постановка задачі:**

**Зчитати символ від користувача.**

**Якщо символ == "к", викликати calculation().**

**Якщо символ == "h", викликати функцію задачі 9.1.**

**Якщо символ == "g", викликати функцію задачі 9.2.**

**Якщо символ == "f", викликати функцію задачі 9.3.**

**Якщо символ інший, видати звуковий сигнал.**

**Запитати користувача про продовження або завершення.**

**Якщо користувач вводить "d", "D" або "с", завершити програму.**

**Інакше, повторити програму.**

**План реалізації:**

**Використати умовні оператори для визначення дії на основі введеного символу.**

**Викликати відповідну функцію або видати звуковий сигнал.**

**Запитати користувача про продовження або завершення.**

**Повторити програму, якщо користувач не вибрав завершення.**

Лістинг коду програми:

#include <iostream>

#include <iomanip>

#include <limits>

#include "Modules/Sumar.h"

using namespace std;

void depositCalculation() {

double depositAmount;

int termMonths;

cout << "Задача № 9.1" << endl;

cout << "Програма для обрахунку суми щомісячних виплат депозиту" << endl;

cout << "Введіть суму депозиту (грн.): ";

cin >> depositAmount;

cout << "Введіть строк дії угоди (місяців): ";

cin >> termMonths;

if (termMonths <= 0 || depositAmount <= 0) {

cout << "Некоректні вхідні дані. Сума депозиту та строк угоди повинні бути додатними." << endl;

return;

}

double annualRate = (termMonths <= 6) ? 16.0 : 18.0;

double monthlyRate = annualRate / 12.0 / 100.0;

double monthlyInterest = depositAmount \* monthlyRate;

double totalInterest = monthlyInterest \* termMonths;

cout << fixed << setprecision(2);

cout << "Сума щомісячних виплат відсотків: " << monthlyInterest << " грн." << endl;

cout << "Загальна сума відсотків за весь строк: " << totalInterest << " грн." << endl;

}

int getBeaufortScale(double windSpeed) {

if (windSpeed < 0.3) return 0;

else if (windSpeed <= 1.5) return 1;

else if (windSpeed <= 3.3) return 2;

else if (windSpeed <= 5.4) return 3;

else if (windSpeed <= 7.9) return 4;

else if (windSpeed <= 10.7) return 5;

else if (windSpeed <= 13.8) return 6;

else if (windSpeed <= 17.1) return 7;

else if (windSpeed <= 20.7) return 8;

else if (windSpeed <= 24.4) return 9;

else if (windSpeed <= 28.4) return 10;

else if (windSpeed <= 32.6) return 11;

else return 12;

}

void windSpeedBeaufort() {

cout << "Задача № 9.2" << endl;

cout << "Програма для обрахунку швидкості вітру в балах Бофорта в січні за масивом середніх даних" << endl;

double windSpeeds[31] = {

2.5, 3.8, 4.2, 5.1, 6.0, 7.2, 4.8, 3.5, 2.8, 5.5,

6.3, 8.1, 9.0, 7.5, 4.0, 3.2, 2.9, 5.8, 6.5, 7.0,

8.5, 9.2, 10.5, 11.0, 8.8, 7.3, 6.2, 5.0, 4.5, 3.9,

2.7

};

string beaufortDescriptions[13] = {

"Безвітря. Дим піднімається вертикально.",

"Дим гнеться, прапори не обертаються.",

"Рух повітря відчувається обличчям. Листя шелестить. Прапори обертаються.",

"Тріпоче листя, хитаються дрібні гілки. Маленькі прапори.",

"Хитаються тонкі гілки дерев. Вітер піднімає пил та шматки паперу.",

"Хитаються великі гілки. На воді з’являються хвилі.",

"Хитаються великі гілки.",

"Хитаються невеликі стовбури дерев. На морі з’являються хвилі, пінитися.",

"Ламаються гілки дерев, важко йти проти вітру.",

"Гілки руйнуються. Зриває покрівлі, руйнує димарі.",

"Гілки руйнуються. Дерева вириваються з корінням.",

"Великі руйнування.",

"Призводить до суттєвих руйнувань."

};

double maxWindSpeed = windSpeeds[0];

int maxBeaufort = getBeaufortScale(windSpeeds[0]);

int maxWindSpeedDay = 1;

cout << "Денні швидкості вітру за січень 2025 року:\n";

for (int day = 0; day < 31; day++) {

int beaufort = getBeaufortScale(windSpeeds[day]);

cout << day + 1 << " січня: Швидкість вітру " << windSpeeds[day]

<< " м/с, Шкала Бофорта " << beaufort << "\n";

if (windSpeeds[day] > maxWindSpeed) {

maxWindSpeed = windSpeeds[day];

maxBeaufort = beaufort;

maxWindSpeedDay = day + 1;

}

}

cout << "\nНайбільша зафіксована швидкість вітру: " << maxWindSpeed

<< " м/с, Шкала Бофорта " << maxBeaufort << "\n";

cout << "Це сталося " << maxWindSpeedDay << " січня.\n";

cout << "Опис: " << beaufortDescriptions[maxBeaufort] << "\n";

}

void binaryCount() {

cout << "Задача № 9.3" << endl;

cout << "Програма для підрахунку двійкових нулів або одиниць залежно від біта D13" << endl;

long long number;

cout << "Введіть натуральне число (від 0 до 9008000): ";

cin >> number;

if (number < 0 || number > 9008000) {

cout << "Некоректне число. Число повинно бути в межах від 0 до 9008000." << endl;

return;

}

bool isD13Set = (number & (1LL << 13)) != 0;

int zeros = 0, ones = 0;

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

if (number & (1LL << i)) {

ones++;

} else {

zeros++;

}

}

int result = isD13Set ? ones : zeros;

cout << "Біт D13 дорівнює " << (isD13Set ? "1" : "0") << ".\n";

cout << "Результат: " << (isD13Set ? "кількість двійкових одиниць: " : "кількість двійкових нулів: ")

<< result << "\n";

}

void calculation() {

cout << "Обчислення функції s\_calculation(x, y, z)" << endl;

cout << "Введіть значення x: ";

double x;

cin >> x;

cout << "Введіть значення y: ";

double y;

cin >> y;

cout << "Введіть значення z: ";

double z;

cin >> z;

cout << fixed << setprecision(2);

cout << "Результат обчислення s\_calculation(" << x << ", " << y << ", " << z << "): " << s\_calculation(x, y, z) << endl;

}

int main() {

system("chcp 65001 > nul");

char choice, exitChoice;

do {

cout << "\nВиберіть задачу для виконання:\n";

cout << "k - Обчислення функції s\_calculation()\n";

cout << "h - Задача 9.1 (обчислення депозиту)\n";

cout << "g - Задача 9.2 (швидкість вітру за шкалою Бофорта)\n";

cout << "f - Задача 9.3 (підрахунок двійкових нулів/одиниць)\n";

cout << "Введіть символ: ";

cin >> choice;

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

switch (choice) {

case 'k':

calculation();

break;

case 'h':

depositCalculation();

break;

case 'g':

windSpeedBeaufort();

break;

case 'f':

binaryCount();

break;

default:

cout << "Некоректний символ! Спробуйте ще раз.\a" << endl;

break;

}

cout << "\nВведіть 'd', 'D' або 'с' для виходу, або будь-який інший символ для продовження: ";

cin >> exitChoice;

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

} while (exitChoice != 'd' && exitChoice != 'D' && exitChoice != 'с');

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

return 0;

}

# Лістинг коду з TestDriver:

#include <iostream>

#include <iomanip>

#include <cmath>

#include "Functions.h"

using namespace std;

void testDepositCalculation() {

cout << "Тестування depositCalculation (Задача 9.1)\n";

DepositResult result1 = depositCalculation(1000.0, 3);

cout << "Тест #1: depositAmount = 1000, termMonths = 3\n";

cout << "Очікуваний monthlyInterest: 13.33, totalInterest: 40.00\n";

cout << "Результат: monthlyInterest = " << result1.monthlyInterest << ", totalInterest = " << result1.totalInterest << "\n";

if (result1.valid && abs(result1.monthlyInterest - 13.33) < 0.01 && abs(result1.totalInterest - 40.00) < 0.01) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

DepositResult result2 = depositCalculation(-100.0, 3);

cout << "Тест #2: depositAmount = -100, termMonths = 3\n";

cout << "Очікується: невалідний результат\n";

cout << "Результат: " << (result2.valid ? "валідний" : "невалідний") << "\n";

if (!result2.valid) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

cout << "-------------------------\n";

}

void testGetBeaufortScale() {

cout << "Тестування getBeaufortScale (допоміжна фуsнкція для Задачі 9.2)\n";

int result1 = getBeaufortScale(2.5);

cout << "Тест #1: windSpeed = 2.5\n";

cout << "Очікуваний результат: 2\n";

cout << "Результат: " << result1 << "\n";

if (result1 == 2) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

int result2 = getBeaufortScale(35.0);

cout << "Тест #2: windSpeed = 35.0\n";

cout << "Очікуваний результат: 12\n";

cout << "Результат: " << result2 << "\n";

if (result2 == 12) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

cout << "-------------------------\n";

}

void testWindSpeedBeaufort() {

cout << "Тестування windSpeedBeaufort (Задача 9.2)\n";

double windSpeeds[31] = {

2.5, 3.8, 4.2, 5.1, 6.0, 7.2, 4.8, 3.5, 2.8, 5.5,

6.3, 8.1, 9.0, 7.5, 4.0, 3.2, 2.9, 5.8, 6.5, 7.0,

8.5, 9.2, 10.5, 11.0, 8.8, 7.3, 6.2, 5.0, 4.5, 3.9,

2.7

};

string beaufortDescriptions[13] = {

"Безвітря. Дим піднімається вертикально.",

"Дим гнеться, прапори не обертаються.",

"Рух повітря відчувається обличчям. Листя шелестить. Прапори обертаються.",

"Тріпоче листя, хитаються дрібні гілки. Маленькі прапори.",

"Хитаються тонкі гілки дерев. Вітер піднімає пил та шматки паперу.",

"Хитаються великі гілки. На воді з’являються хвилі.",

"Хитаються великі гілки.",

"Хитаються невеликі стовбури дерев. На морі з’являються хвилі, пінитися.",

"Ламаються гілки дерев, важко йти проти вітру.",

"Гілки руйнуються. Зриває покрівлі, руйнує димарі.",

"Гілки руйнуються. Дерева вириваються з корінням.",

"Великі руйнування.",

"Призводить до суттєвих руйнувань."

};

WindSpeedResult result = windSpeedBeaufort(windSpeeds, beaufortDescriptions);

cout << "Тест #1: Використання масиву windSpeeds\n";

cout << "Очікуваний maxWindSpeed: 11.0, maxBeaufort: 6, maxWindSpeedDay: 24\n";

cout << "Результат: maxWindSpeed = " << result.maxWindSpeed << ", maxBeaufort = " << result.maxBeaufort << ", maxWindSpeedDay = " << result.maxWindSpeedDay << "\n";

if (result.valid && abs(result.maxWindSpeed - 11.0) < 0.01 && result.maxBeaufort == 6 && result.maxWindSpeedDay == 24) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

cout << "-------------------------\n";

}

void testBinaryCount() {

cout << "Тестування binaryCount (Задача 9.3)\n";

BinaryCountResult result1 = binaryCount(8192);

cout << "Тест #1: number = 8192\n";

cout << "Очікуваний результат: isD13Set = true, result = 1 (кількість одиниць)\n";

cout << "Результат: isD13Set = " << (result1.isD13Set ? "true" : "false") << ", result = " << result1.result << "\n";

if (result1.valid && result1.isD13Set && result1.result == 1) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

BinaryCountResult result2 = binaryCount(9008001);

cout << "Тест #2: number = 9008001\n";

cout << "Очікується: невалідний результат\n";

cout << "Результат: " << (result2.valid ? "валідний" : "невалідний") << "\n";

if (!result2.valid) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

cout << "-------------------------\n";

}

void testCalculation() {

cout << "Тестування calculation (Задача 9.4)\n";

double result1 = calculation(1.0, 2.0, 3.0);

cout << "Тест #1: x = 1, y = 2, z = 3\n";

cout << "Очікуваний результат: -5.48\n";

cout << "Результат: " << result1 << "\n";

if (abs(result1 - (-5.48)) < 0.01) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

double result2 = calculation(0.0, 0.0, 0.0);

cout << "Тест #2: x = 0, y = 0, z = 0\n";

cout << "Очікуваний результат: 0.00\n";

cout << "Результат: " << result2 << "\n";

if (abs(result2 - 0.0) < 0.01) {

cout << "Тест пройдено!\n";

} else {

cout << "Тест не пройдено!\n";

}

cout << "-------------------------\n";

}

int main() {

system("chcp 65001 > nul");

cout << "Запуск TestDriver для всіх функцій\n\n";

testDepositCalculation();

testGetBeaufortScale();

testWindSpeedBeaufort();

testBinaryCount();

testCalculation();

cout << "Тестування завершено.\n";

system("pause");

return 0;

}

# Відповіді на контрольні запитання

1. **Яким оператором С/С++ можливо повноцінно замінити тернарний оператор?**  
   Умовним оператором if-else. Можна експериментально показати, що ви отримуєте той самий результат, використовуючи обидва підходи.
2. **Що в програмуванні розуміють під пріоритетом виконання операцій та асоціативністю?**  
   Пріоритет визначає порядок обчислення операцій, а асоціативність – напрямок виконання операцій з однаковим пріоритетом (зліва направо або справа наліво).
3. **Яку область видимості мають змінні, оголошені в тілі циклу або умови?**  
   Такі змінні мають блочну область видимості (локальні для цього блоку).
4. **Якою є асоціативність операцій (арифметичних, логічних, побітових, інкремент/декремент, тернарного, порівняння) в С/С++?**

Наприклад, операції множення/ділення мають вищий пріоритет, а тернарний оператор має праву асоціативність.

1. **Перелічіть випадки, за яких доцільно використовувати тернарний оператор С/С++**  
   Коли потрібно компактно записати умовний вираз, наприклад:

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

1. **Яке значення міститиме змінна cnt після виконання наступної інструкції: cnt--;**  
   cnt буде зменшено на 1.
2. **Чим константна змінна відрізняється від звичайної? Коли її використовувати?**  
   Константна змінна (з кваліфікатором const) не може бути змінена після ініціалізації, що покращує безпеку коду. Її використовують, коли значення не повинно змінюватися.
3. **Яких типів можуть бути операнди логічних операторів?**  
   У С/С++ операнди логічних операторів повинні бути приведені до типу bool.
4. **Яке значення міститиме змінна cnt при: bool cnt = !!0;**  
   cnt буде false (тобто 0), оскільки 0 перетворюється в false.
5. **Сформулюйте правило запису виразу ініціалізації у циклах з параметром (for)**  
   Вираз ініціалізації в циклі for виконується один раз перед початком циклу і визначає початкове значення лічильника або інших змінних.

**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.