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

Авраменко В. В.

ПЕРЕВІРИВ

викладач кафедри кібербезпеки   
та програмного забезпечення

Коваленко А. С.

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.

**Варіант №19**

**Завдання до лабораторної роботи**

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

## **ЗАДАЧА 9.1**

### ****1. Аналіз і постановка задачі****

**Мета:** визначити ступінь хвилювання моря (у балах) за заданою висотою хвиль у метрах.

**Вхідні дані:** дійсне число – висота хвиль у метрах.

**Вихідні дані:** ціле число від 0 до 9 – бал хвилювання моря відповідно до таблиці.

### ****2. Аналіз вимог****

* Програма має обробляти значення висоти хвиль з плаваючою комою.
* Необхідна точна відповідність діапазонам згідно таблиці.
* Вивід має бути числом (0–9), опціонально з описом терміну (наприклад, “ROUGH”).

### ****3. Проєктування архітектури****

* Проста консольна програма.
* Один вхід – число з плаваючою крапкою.
* Умовна конструкція if-elif-else або switch.

### ****4. Детальне проектування модулів****

**Функція** int wave\_grade(float h):

* Приймає висоту хвилі.
* Повертає оцінку хвилювання моря (0–9).

## **ЗАДАЧА 9.2**

### ****1. Аналіз і постановка задачі****

**Мета:** перетворити температуру з Фаренгейта в Цельсій.

**Вхідні дані:** дійсне число – температура у градусах Фаренгейта.

**Вихідні дані:** дійсне число – температура у градусах Цельсія.

### ****2. Аналіз вимог****

* Просте обчислення за відомою формулою.
* Необхідна правильність розрахунку з урахуванням десяткової точності.

### ****3. Проєктування архітектури****

* Одна функція, що приймає та повертає .
* Консольна програма з виводом результату.

### ****4. Детальне проектування модулів****

**Функція** float fahrenheit\_to\_celsius(float t\_f):

* Реалізує формулу: .

## **ЗАДАЧА 9.3**

### ****1. Аналіз і постановка задачі****

**Мета:** залежно від значення біта D₁ (другий справа, тобто біт №1), або підрахувати кількість бінарних нулів, або суму одиниць у двійковому представленні.

**Вхідні дані:** ціле число N від 0 до 65535.

**Вихідні дані:**

* якщо біт D₁ == 1 → рахуємо **кількість нулів**;
* якщо D₁ == 0 → рахуємо **суму одиниць**.

### ****2. Аналіз вимог****

* Потрібна побітова операція для визначення значення біта.
* Потрібна бінарна репрезентація числа.
* Для підрахунку – використовуємо тернарний оператор.

### ****3. Проєктування архітектури****

* Отримання біта через побітову операцію: (N >> 1) & 1
* Далі залежно від значення – рахуємо або нулі, або одиниці в бінарному представленні (bin(N)).

### ****4. Детальне проектування модулів****

**Функція** int process\_binary(int N):

* bit = (N >> 1) & 1
* result = bin(N)[2:]
* Якщо bit == 1: рахуємо кількість '0' у result
* Інакше – суму одиниць: sum(int(b) for b in result)

# Тестові набори

**Задача 9.1 – Оцінка хвилювання моря**

**Опис:** За заданою висотою хвилі (у метрах) визначити бал хвилювання згідно таблиці.  
**Контрольні приклади:**

* **Тест 1 (крайова ситуація – мінімальне значення):**
  + **Вхід:** 0.0
  + **Очікуваний вихід:** 0  
    *(0 м → CALM-GLASSY)*
* **Тест 2 (нижня межа наступного діапазону):**
  + **Вхід:** 0.1
  + **Очікуваний вихід:** 1  
    *(0.1 м знаходиться на межі CALM-RIPPLED)*
* **Тест 3 (середина діапазону):**
  + **Вхід:** 0.3
  + **Очікуваний вихід:** 2  
    *(0.3 м → SMOOTH-WAVELET)*
* **Тест 4 (середнє значення іншого діапазону):**
  + **Вхід:** 1.0
  + **Очікуваний вихід:** 3  
    *(1.0 м → SLIGHT)*
* **Тест 5 (верхня межа середніх значень):**
  + **Вхід:** 2.0
  + **Очікуваний вихід:** 4  
    *(2.0 м → MODERATE)*
* **Тест 6 (середнє значення для діапазону “ROUGH”):**
  + **Вхід:** 3.0
  + **Очікуваний вихід:** 5  
    *(3.0 м → ROUGH)*
* **Тест 7 (для “VERY ROUGH”):**
  + **Вхід:** 5.0
  + **Очікуваний вихід:** 6  
    *(5.0 м → VERY ROUGH)*
* **Тест 8 (для “HIGH”):**
  + **Вхід:** 7.0
  + **Очікуваний вихід:** 7  
    *(7.0 м → HIGH)*
* **Тест 9 (для “VERY HIGH”):**
  + **Вхід:** 10.0
  + **Очікуваний вихід:** 8  
    *(10.0 м → VERY HIGH)*
* **Тест 10 (для “PHENOMENAL”):**
  + **Вхід:** 15.0
  + **Очікуваний вихід:** 9  
    *(15.0 м більше 14 м → PHENOMENAL)*

**Задача 9.2 – Конвертація температури з Фаренгейта у Цельсій**

**Опис:** Перевести температуру за шкалою Фаренгейта в температуру за шкалою Цельсія за формулою:

**Контрольні приклади:**

* **Тест 1 (крайова точка замерзання води):**
  + **Вхід:** 32°F
  + **Очікуваний вихід:** 0°C
* **Тест 2 (точка кипіння води):**
  + **Вхід:** 212°F
  + **Очікуваний вихід:** 100°C
* **Тест 3 (помірна температура):**
  + **Вхід:** 68°F
  + **Очікуваний вихід:** 20°C
* **Тест 4 (негативна температура):**
  + **Вхід:** -40°F
  + **Очікуваний вихід:** -40°C  
    *(Оскільки -40°F дорівнює -40°C)*
* **Тест 5 (типове значення):**
  + **Вхід:** 100°F
  + **Очікуваний вихід:** приблизно 37.78°C

**Задача 9.3 – Обчислення за двійковим представленням числа**

**Опис:**  
Для натурального числа NN (від 0 до 65535) потрібно перевірити, чи рівний другий справа біт (D₁) 1.

* Якщо D₁ = 1, то результат – кількість двійкових нулів у представленні числа.
* Якщо D₁ = 0, то результат – сума (кількість) двійкових одиниць у представленні.

**Контрольні приклади:**

* **Тест 1 (D₁ = 1):**
  + **Вхід:** 2
  + **Бінарне представлення:** "10"
  + **Перевірка:** (2>>1)&1=1(2 >> 1) \& 1 = 1
  + **Очікуваний вихід:** 15 (кількість '0' = 15)
* **Тест 2 (D₁ = 0):**
  + **Вхід:** 1
  + **Бінарне представлення:** "1" (або "0000000000000001" при повному представленні)
  + **Перевірка:** (1>>1)&1=0(1 >> 1) \& 1 = 0
  + **Очікуваний вихід:** 1 (сума одиниць = 1)
* **Тест 3 (D₁ = 0):**
  + **Вхід:** 9
  + **Бінарне представлення:** "1001"
  + **Перевірка:** (9>>1)&1=0(9 >> 1) \& 1 = 0 (оскільки "1001" >> 1 = "100", останній біт = 0)
  + **Очікуваний вихід:** 2 (кількість одиниць у "1001" = 2)
* **Тест 4 (крайній випадок):**
  + **Вхід:** 0
  + **Бінарне представлення:** "0"
  + **Перевірка:** (0>>1)&1=0(0 >> 1) \& 1 = 0
  + **Очікуваний вихід:** 0 (сума одиниць = 0)

# Результат роботи тестового драйверу:

Тестування задачі 9.1

========

Аргумент(-и): 0

Очікується: 0

Отримано: 0 - PASSED

========

Аргумент(-и): 0.1

Очікується: 1

Отримано: 1 - PASSED

========

Аргумент(-и): 0.3

Очікується: 2

Отримано: 2 - PASSED

========

Аргумент(-и): 1

Очікується: 3

Отримано: 3 - PASSED

========

Аргумент(-и): 2

Очікується: 4

Отримано: 4 - PASSED

========

Аргумент(-и): 3

Очікується: 5

Отримано: 5 - PASSED

========

Аргумент(-и): 5

Очікується: 6

Отримано: 6 - PASSED

========

Аргумент(-и): 7

Очікується: 7

Отримано: 7 - PASSED

========

Аргумент(-и): 10

Очікується: 8

Отримано: 8 - PASSED

========

Аргумент(-и): 15

Очікується: 9

Отримано: 9 - PASSED

========

Тестування задачі 9.2

========

Аргумент(-и): 32

Очікується: 0

Отримано: 0 - PASSED

========

Аргумент(-и): 212

Очікується: 100

Отримано: 100 - PASSED

========

Аргумент(-и): 68

Очікується: 20

Отримано: 20 - PASSED

========

Аргумент(-и): -40

Очікується: -40

Отримано: -40 - PASSED

========

Аргумент(-и): 100

Очікується: 37.78

Отримано: 37.7778 - PASSED

========

Тестування задачі 9.3

========

Аргумент(-и): 2

Очікується: 15

Отримано: 15 - PASSED

========

Аргумент(-и): 1

Очікується: 1

Отримано: 1 - PASSED

========

Аргумент(-и): 9

Очікується: 2

Отримано: 2 - PASSED

========

Аргумент(-и): 0

Очікується: 0

Отримано: 0 - PASSED

========

# **ЗАДАЧА 9.4**

## Аналіз вимог до ПЗ

* **Вхід користувача:**  
  Програма повинна очікувати введення символу з клавіатури.
* **Обробка символів:**
  + Якщо користувач вводить:
    - 'j'**:** викликається функція s\_calculation().
    - 'z'**:** викликається функція задачі 9.1 (оцінка хвилювання моря).
    - 'х'**:** викликається функція задачі 9.2 (конвертація температури).
    - 'c'**:** викликається функція задачі 9.3 (обчислення за бінарним представленням).
  + Якщо введено будь-який інший символ – такий ввід ігнорується, користувачу видається звуковий сигнал (beep) або повідомлення про помилкове введення.
* **Логіка завершення роботи:**  
  Після виконання операції програма запитує у користувача:
  + Якщо введено символ 'v', 'V' або 'А' – відбувається вихід з програми.
  + Інакше – програма повторює цикл вводу та обробки.

**Основний алгоритм (псевдокод):**

Поки (true)

Вивести запит на введення символу для виконання операції

Зчитати символ input

Перемикач (input)

випадок 'j':

виклик s\_calculation()

випадок 'z':

виклик функції задачі 9.1

випадок 'х':

виклик функції задачі 9.2

випадок 'c':

виклик функції задачі 9.3

за замовчуванням:

видаємо звуковий сигнал/повідомлення про помилкове введення

Вивести запит: "Для виходу введіть 'v', 'V' або 'А'. Для продовження – інший символ"

Зчитати символ exitChar

Якщо exitChar є 'v' або 'V' або 'А', тоді break (вихід з циклу)

Кінець циклу