

**Мета роботи** полягає у набутті ґрунтовних вмінь і практичних навичок реалізації у Code::Blocks IDE мовою програмування С++ програмних модулів створення й оброблення даних типів масив, структура, об’єднання, множина, перелік, перетворення типів даних, використання файлових потоків та функцій стандартних бібліотек для оброблення символьної інформації.

**Завдання:**

1. Реалізувати програмні модулі розв’язування задач 10.1–10.3 як складові статичної бібліотеки libModulesПрізвище.а (проект ModulesПрізвище лабораторних робіт №8–9).

2. Реалізувати тестовий драйвер автоматизованої перевірки програмних модулів розв’язування задач 10.1–10.3.

**Варіант №14:**

1. **Аналіз задачі 10.1:**

**Мета**: аналіз вхідного текстового файлу та створення узагальненої інформації на основі його вмісту.

Необхідно реалізувати:

* Запис авторської інформації: ім’я, прізвище розробника, організація, місто, країна, рік розробки.
* Витягти та записати всі символи верхнього регістру з вхідного файлу.
* Проаналізувати, чи всі рядки завершуються розділовими знаками (як у вірші), й записати відповідне повідомлення.
* Весь вірш переписати в один стовпчик по 4 рядки.

**Архітектурне проектування 10.1:**

#### Вхідні дані:

input.txt — вхідний файл з віршем.

#### Вихідні дані:

output.txt — файл з результатами аналізу тексту.

#### Функції:

getAuthorInfo() — повертає авторську інформацію.

extractUppercase(fileName) — отримує всі символи верхнього регістру.

checkPunctuation(fileName) — перевіряє, чи кожен рядок закінчується розділовим знаком.

writePoemVertically(fileName) — записує вірш в один стовпчик по 4 рядки.

**Псевдокод до задачі 10.1:**

Відкрити input.txt для читання

Відкрити output.txt для запису (очистити, якщо існує)

1. Авторська інформація:

автор ← getAuthorInfo()

записати автор у output.txt

2. Великі літери:

upperLetters ← extractUppercase("input.txt")

записати upperLetters у output.txt

3. Перевірка розділових знаків:

punctuationResults ← checkPunctuation("input.txt")

записати punctuationResults у output.txt

4. Запис вірша стовпчиком:

writePoemVertically("input.txt", "output.txt")

Закрити обидва файли

1. **Аналіз задачі 10.2:**

**Мета:** доповнення вихідного файлу мета-інформацією.

Необхідно дописати:

* Розмір файлу у кілобайтах (КБ).
* Дату і час модифікації файлу — момент виконання програми.

### **Архітектурне проєктування 10.2:**

#### Вхідні дані:

output.txt — файл після виконання задачі 10.1.

#### Вихідні дані:

output.txt з дописаними метаданими.

#### Функції:

getFileSizeKB(fileName) — обчислює розмір файлу в КБ.

getCurrentDateTime() — отримує дату і час останньої модифікації.

**Псевдокод до задачі 10.2:**

Відкрити output.txt у режимі додавання

1. Отримати розмір файлу:

size ← getFileSizeKB("output.txt")

записати "Розмір файлу: size КБ" у output.txt

2. Отримати дату і час:

dateTime ← getCurrentDateTime()

записати "Дата і час: dateTime" у output.txt

Закрити output.txt

1. **Аналіз задачі 10.3:**

**Мета**: обробка вхідних числових даних та їх аналітичне представлення у вихідному файлі.

Необхідно реалізувати:

* Вхідні дані: цілі значення x, y, z та натуральне b.
* Виклик функції s\_calculation(x, y, z) із заголовкового файлу та запис результату.
* Представлення числа y у двійковій формі (переведення з десяткової системи).

### **Архітектурне проєктування 10.3:**

#### Вхідні дані:

Числові значення: x, y, z, b.

#### Вихідні дані:

output.txt з результатами обчислення і двійковим поданням y.

#### Функції:

s\_calculation(x, y, z) — обчислення з зовнішнього заголовкового файлу Modules/Kondratenko.h.

toBinary(y) — переведення числа y у двійкову систему числення.

**Псевдокод до 10.3:**

Ввід x, y, z, b (з клавіатури або у коді)

Відкрити output.txt у режимі додавання

1. Обчислення:

результат ← s\_calculation(x, y, z)

записати "Результат обчислення: результат" у output.txt

2. Подання y у двійковій системі:

yBinary ← toBinary(y)

записати "Число y в двійковій формі: yBinary" у output.txt

Закрити output.txt

Тест-кейси до задач:

Додаток А:



Додаток Б:

Додаток В:



Лістинг ModulesKondratenko:

#include <iostream>

#include <cmath>

#include <bitset>

#include <fstream>

#include <string>

#include <ctime>

#include <cctype>

using namespace std;

double s\_calculation(double x, double y, double z){

return fabs(abs(sin(y - pow(z, 2 ))) + sqrt(x) - sqrt(pow((y \* z), x) + y / (2 \* M\_PI)));

}

double salary\_calculation(double salary, int experience, double office\_salary){

double bonus = 0.0;

if(experience >= 3 && experience <= 5)

bonus = office\_salary \* 0.10;

else if(experience >= 5 && experience <=7)

bonus = office\_salary \* 0.15;

else if(experience >= 7 && experience <= 15)

bonus = office\_salary \* 0.20;

else if(experience > 15)

bonus = office\_salary \* 0.25;

return salary + bonus;

}

void temperature\_calculation(const double temps[], int size, double& avg\_celsius, double& avg\_fahrenheit){

double sum = 0.0;;

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

sum += temps[i];

avg\_celsius = sum / size;

avg\_fahrenheit = 32 + (9.0/5.0) \*avg\_celsius;

}

double analyze\_bit(unsigned int N){

int bit11 = (N >> 11) & 1;

int count = 0;

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

count += (((N >> i) & 1) == bit11) ? 1 : 0;

}

return count;

}

string getAuthorInfo() {

return "Розробник: Кондратенко Дмитро, Центральноукраїнський Національний Технічний Університет, Кропивницький, Україна, 2025\n";

}

string extractUppercase(const string& filename) {

ifstream file(filename);

string line, result;

while (getline(file, line)) {

for (char ch : line) {

if (isupper(static\_cast<unsigned char>(ch))) {

result += ch;

}

}

}

return "Великі літери з тексту: " + result + "\n";

}

string checkPunctuation(const string& filename) {

ifstream file(filename);

string line;

int lineNum = 0;

string result = "Перевірка розділових знаків:\n";

while (getline(file, line)) {

lineNum++;

char lastChar = line.empty() ? ' ' : line.back();

if (lastChar == '.' || lastChar == ',' || lastChar == ';' || lastChar == ':' || lastChar == '!' || lastChar == '?') {

result += "Рядок " + to\_string(lineNum) + ": закінчується розділовим знаком\n";

} else {

result += "Рядок " + to\_string(lineNum) + ": НЕ закінчується розділовим знаком\n";

}

}

return result + "\n";

}

void writePoemVertically(const string& inputFile, const string& outputFile) {

ifstream file(inputFile);

string lines[4];

for (int i = 0; i < 4 && getline(file, lines[i]); i++) {}

ofstream out(outputFile, ios::app);

out << "Вірш у стовпчик:\n";

size\_t maxLen = 0;

for (const auto& l : lines) maxLen = max(maxLen, l.length());

for (size\_t i = 0; i < maxLen; i++) {

for (int j = 0; j < 4; j++) {

if (i < lines[j].length()) {

out << lines[j][i] << " ";

} else {

out << " ";

}

}

out << "\n";

}

}

string getFileSizeKB(const string& filename) {

ifstream in(filename, ios::binary | ios::ate);

streamsize size = in.tellg();

in.close();

return "Розмір файлу: " + to\_string(size / 1024.0) + " KB\n";

}

string getCurrentDateTime() {

time\_t now = time(0);

string dt = ctime(&now);

return "Дата і час: " + dt + "\n";

}

string toBinary(int b) {

return bitset<16>(b).to\_string();

}

Лістинг ModulesKondratenko.h:

#ifndef MODULESKONDRATENKO\_H\_INCLUDED

#define MODULESKONDRATENKO\_H\_INCLUDED

#include <cmath>

#include <string>

using namespace std;

double s\_calculation(double x, double y, double z);

double salary\_calculation(double salary, int experience, double office\_salary);

void temperature\_calculation(const double temps[], int size, double& avg\_celsius, double& avg\_fahrenheit);

double analyze\_bit(unsigned int N);

string getAuthorInfo();

string extractUppercase(const string& filename);

string checkPunctuation(const string& filename);

void writePoemVertically(const string& inputFile, const string& outputFile);

string getFileSizeKB(const string& filename);

string getCurrentDateTime();

string toBinary(int b);

#endif // MODULESKONDRATENKO\_H\_INCLUDED

Лістинг TestDriver:

#include <iostream>

#include <fstream>

#include <string>

#include <cassert>

#include <cmath>

#include "../../../lab08/prj/ModulesKondratenko/ModulesKondratenko.h"

using namespace std;

void test\_10\_1() {

cout << "\n===== Тест 10.1: Робота з текстом =====\n";

ofstream input("input.txt");

input << "Ми Летимо Увись,\n"

<< "Де Сяє Сонце.\n"

<< "Світло Пронизує Небо,\n"

<< "І Вітер Шепоче.\n";

input.close();

cout << "[INFO] Вхідний файл input.txt створено.\n";

ofstream clearOut("output.txt");

clearOut.close();

cout << "[INFO] Вихідний файл output.txt очищено.\n";

ofstream out("output.txt", ios::app);

string author = getAuthorInfo();

string uppercase = extractUppercase("input.txt");

string punctuation = checkPunctuation("input.txt");

cout << "[] Авторська інформація:\n" << author;

cout << "[] Великі літери: " << uppercase;

cout << "[] Перевірка пунктуації:\n" << punctuation;

out << author;

out << uppercase;

out << punctuation;

out.close();

writePoemVertically("input.txt", "output.txt");

cout << "[INFO] Вірш записано у вертикальному вигляді.\n";

cout << "[Test 10.1] Успішно завершено\n";

}

void test\_10\_2() {

cout << "\n===== Тест 10.2: Запис розміру та часу =====\n";

string fileSize = getFileSizeKB("output.txt");

string dateTime = getCurrentDateTime();

cout << " Розмір файлу: " << fileSize;

cout << " Дата і час: " << dateTime;

ofstream out("output.txt", ios::app);

out << fileSize;

out << dateTime;

out.close();

cout << "[Test 10.2] Успішно завершено\n";

}

void test\_10\_3() {

cout << "\n===== Тест 10.3: Обчислення формули та двійкова система =====\n";

struct TestCase {

int x, y, z, b;

};

TestCase test1 = {4, 2, 3, 2};

TestCase test2 = {16, 3, 1, 8};

TestCase test3 = {9, 1, 2, 1};

ofstream out("output.txt", ios::app);

double result1 = s\_calculation(test1.x, test1.y, test1.z);

string binary1 = toBinary(test1.b);

cout << "Test 1: s\_calculation(" << test1.x << ", " << test1.y << ", " << test1.z << ") = " << result1 << endl;

cout << " toBinary(" << test1.b << ") = " << binary1 << endl;

out << "Test 1:\nРезультат обчислення: " << result1 << "\nЧисло b в двійковій формі: " << binary1 << "\n";

double result2 = s\_calculation(test2.x, test2.y, test2.z);

string binary2 = toBinary(test2.b);

cout << " Test 2: s\_calculation(" << test2.x << ", " << test2.y << ", " << test2.z << ") = " << result2 << endl;

cout << " toBinary(" << test2.b << ") = " << binary2 << endl;

out << "Test 2:\nРезультат обчислення: " << result2 << "\nЧисло b в двійковій формі: " << binary2 << "\n";

double result3 = s\_calculation(test3.x, test3.y, test3.z);

string binary3 = toBinary(test3.b);

cout << " Test 3: s\_calculation(" << test3.x << ", " << test3.y << ", " << test3.z << ") = " << result3 << endl;

cout << " toBinary(" << test3.b << ") = " << binary3 << endl;

out << "Test 3:\nРезультат обчислення: " << result3 << "\nЧисло b в двійковій формі: " << binary3 << "\n";

out.close();

cout << "[Test 10.3] Успішно завершено\n";

}

int main() {

setlocale(LC\_ALL, "Ukrainian");

setlocale(LC\_ALL, "uk\_UA.UTF-8");

cout << "=== Автоматичне тестування Лабораторної №10 ===\n";

test\_10\_1();

test\_10\_2();

test\_10\_3();

cout << "\n Усі тести пройдено успішно!\n";

return 0;

}

Аргументи до виконання лабораторної роботи:

1. Бо без лаби пари не зарахують.
2. Бо буде залік — а без цього до сесії не доживеш.
3. Бо викладач просив.
4. Бо на GitHub потім буде, виглядає як портфоліо.
5. Бо можна вивчити C++ і не відставати.
6. Щоб розібратись із масивами.
7. Щоб знати, що таке struct.
8. Бо з файлами працювати треба вміти.
9. Тестування — важлива штука, десь стане в пригоді.
10. Бо інакше викладач спитає на захисті — а що скажеш?
11. Потрібно вміти писати README, як у справжніх проєктах.
12. Щоб не лякали слова "unit-тест".
13. Розібратись з Git-репозиторієм.
14. Бо в майбутньому ці штуки точно знадобляться.
15. Плюсик у резюме.
16. Бо на співбесіді можуть спитати: "Що ти робив у універі?"
17. Бо самому цікаво (ну або хоча б треба зробити вигляд).
18. Щоб потім не вчитись усе за ніч.
19. Бо можна когось в групі виручити, якщо сам зробиш.
20. Трохи програмування — трохи прокачки мозку.
21. Щоб навчитись шукати помилки в коді.
22. Бо хочеться відчути себе розробником.
23. Для загального розвитку.
24. Бо завдання реальні — не просто теорія.
25. Щоб не соромно було на консультації.
26. Бо знати, як працює Code::Blocks — корисно.
27. Щоб потім не шукати репу від розпачу перед дедлайном.
28. Тест-кейси — звучить серйозно, хай буде.
29. Можна зробити автоматизацію — це круто.
30. Бо треба щось запушити в Git, не просто ж аккаунт створював.
31. Щоб навчитись користуватись функціями із cstring.
32. Бо struct в C++ — це не страшно.
33. Бо інакше буде вічна перездача.
34. Щоб здати без зайвих питань.
35. Бо все одно треба — краще раніше зробити.
36. Бо класно, коли драйвер сам перевіряє, що написав.
37. Бо треба тренуватись оформлювати звіти.
38. Щоб потім простіше було на курсовій.
39. Бо отримаєш досвід реального проєкту.
40. Бо на парі можна сказати: "Я вже все зробив".
41. Щоб батьки пишались.
42. Щоб викладач не "вантажив".
43. Бо зробити самому — кайфово.
44. Можна далі розвинути проєкт — чому б і ні?
45. Бо працювати з кодом – це як тренажер для мозку.
46. Можна потренувати роботу з файлами.
47. Бо кожен модуль – маленька перемога.
48. Навчишся працювати з тестами.
49. Логіка в коді — це весело (іноді ).
50. Бо GPT допоможе зробити, а здавати треба тобі.