Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 10

з навчальної дисципліни

“Базові методології та технології програмування”

Реалізація програмних модулів оброблення даних складових типів з файловим введенням/виведенням

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

[https://github.com/odorenskyi/](https://github.com/odorenskyi/Dmytro-Parkhomenko-KB18)

ВИКОНАВ

студентка академічної групи КБ-24

Селіхова К.С.

https://github.com/Levanori

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Коваленко А. С.

Кропивницький – 2025

**Мета роботи :**

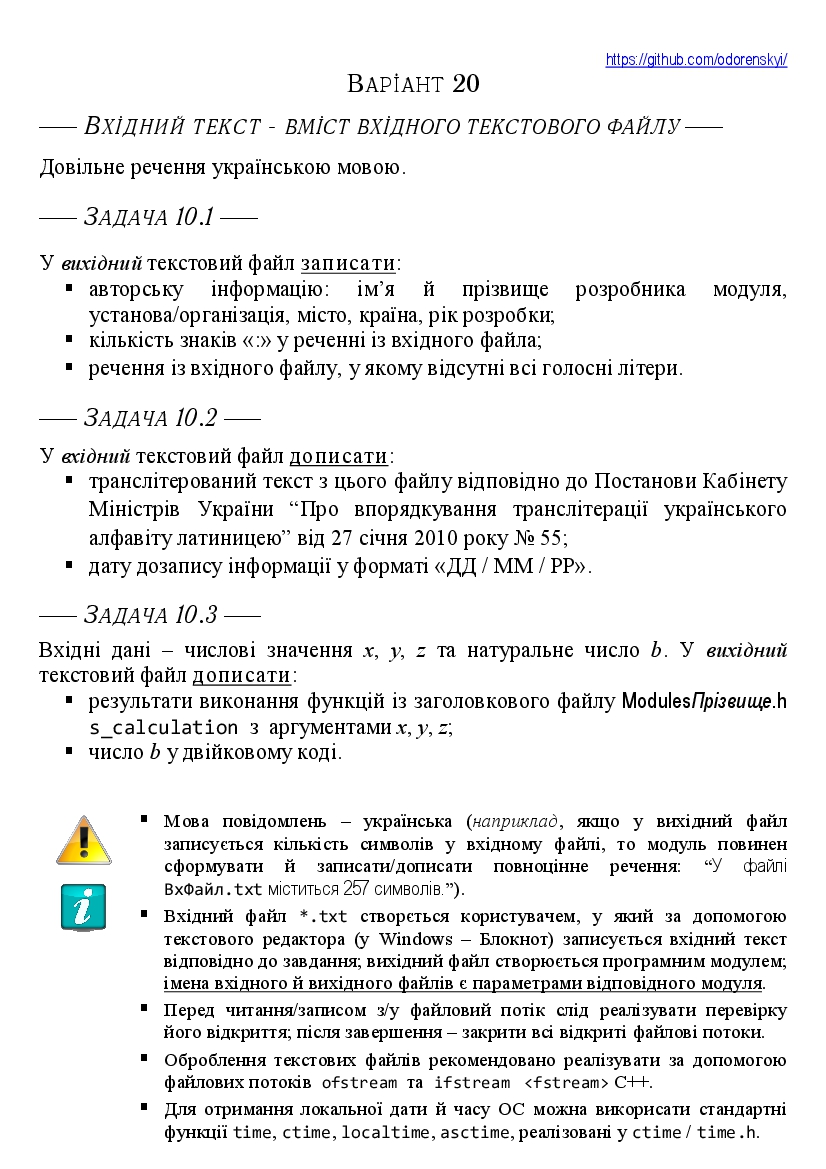
полягає у набутті ґрунтовних вмінь і практичнихнавичок реалізації технології модульного програмування,застосування операторів С/С++ арифметичних, логічних, побітовихоперацій, умови, циклів та вибору під час розроблення статичнихбібліотек, заголовкових файлів та програмних засобів у кросплатформовому середовищі Code::Blocks.

**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.

**BAPİAHT 20**



**Задача 10.1**

Метою першого завдання є створення вихідного текстового файлу, до якого потрібно внести кілька видів інформації, зокрема авторську, статистичну та текстову. Насамперед, студент має самостійно вказати власне ім’я та прізвище, назву установи чи організації, місто, країну й рік розробки модуля. Ця інформація виконує роль титульної частини виводу та надає ідентифікацію програмного продукту.

Далі з тексту вхідного файлу потрібно обробити речення, зокрема знайти й підрахувати кількість символів лапок ("). Ця задача потребує уважного пошуку символів, оскільки лапки можуть бути в різних частинах тексту, і програма повинна проаналізувати рядки послідовно. Наступним етапом є пошук речення, у якому повністю відсутні всі голосні літери українського алфавіту (а, е, є, и, і, ї, о, у, ю, я). Це завдання демонструє навички текстової обробки, зокрема фільтрації рядків за умовами. Результат повинен бути записаний у вихідний файл у вигляді повноцінних речень українською мовою.

Таким чином, задача 10.1 передбачає аналіз текстового вмісту, пошук специфічних символів та побудову граматично коректного виводу. Вона вчить працювати з файлами через потоки ifstream та ofstream, а також перевіряти відкриття/закриття файлів перед обробкою.

**Задача 10.2**

Це завдання полягає в тому, щоб доповнити вхідний текстовий файл новою інформацією. По-перше, весь вміст файлу потрібно транслітерувати, тобто перевести український текст у латиницю відповідно до державного стандарту транслітерації, затвердженого Постановою Кабінету Міністрів України № 55 від 27 січня 2010 року. Для цього слід реалізувати алгоритм або таблицю відповідностей між українськими буквами та їх латинськими аналогами, наприклад: "ж" → "zh", "ч" → "ch", "ш" → "sh", "ю" → "iu" тощо.

Окрім транслітерації, програма повинна автоматично додати поточну дату в кінець файлу. Формат дати вимагається у вигляді "ДД / ММ / РР", тому потрібно скористатися стандартними функціями ctime, localtime або strftime, які дозволяють отримати системну дату у зручному вигляді.

Таким чином, ця задача поєднує роботу з рядками, транслітерацію українського тексту та отримання системної дати. Зміни повинні відбуватись без видалення попереднього вмісту, тобто через **режим дописування (append)**.

**Задача 10.3**

Останнє завдання спрямоване на роботу з числовими даними. Користувач має ввести два числових значення — x і y, а також одне натуральне число B. Після цього потрібно викликати функцію s\_calculation, яка знаходиться в окремому заголовковому файлі Modules/Прізвище.h. Ця функція має бути реалізована користувачем самостійно й може виконувати будь-які математичні обчислення з аргументами x та y (наприклад, добуток, сума, логарифм, синус тощо).

Окрім результатів функції, до вихідного файлу потрібно дописати значення B, представлене у двійковій системі числення. Це означає, що необхідно реалізувати перетворення з десяткової системи у двійкову. Така операція в C++ може бути реалізована вручну через ділення на два або за допомогою бібліотеки bitset.

Також до файлу необхідно додати українське речення, яке повідомляє про кількість символів у вхідному файлі. Наприклад: "У файлі Вхфайл.txt міститься 217 символів." Для цього потрібно відкрити вхідний файл, прочитати його вміст і підрахувати кількість символів, включаючи пробіли й службові символи.

Завдання 10.3 демонструє навички роботи з математичними функціями, власними модулями, перетворенням числових форматів, а також зчитуванням і аналізом вмісту текстових файлів.

**Висновок**

Завдання варіанту 20 охоплюють різні аспекти програмування мовою C++: текстову обробку, транслітерацію, математичні обчислення, роботу з файлами та датою. Особливу увагу приділено валідації відкриття файлів і створенню повноцінного, логічного текстового виводу українською мовою. Рішення кожного пункту вимагає точного дотримання формулювання задачі, використання відповідних стандартних бібліотек (<fstream>, <ctime>, <string>, <bitset>) і демонструє здатність до побудови структурованих програмних модулів. Це завдання також формує практичні навички роботи з файлами в реальних умовах обробки інформації.

Гаразд, зрозумів. Вам потрібні 50 аргументів, які можуть бути використані в контексті захисту або представлення лабораторної роботи. Це можуть бути аргументи, що стосуються важливості роботи, її переваг, складності, використаних підходів, отриманих навичок тощо.

**Аргументи:**

1. Робота демонструє практичне застосування теоретичних знань.
2. Вирішує реальну або типову задачу обробки даних.
3. Освітлює актуальну проблему обробки текстової інформації.
4. Сприяє розвитку навичок роботи з файлами, що є базовим для програміста.
5. Показує розуміння особливостей обробки української мови (локаль, кодування).
6. Акцентує увагу на важливості транслітерації в міжнародному контексті.
7. Використовує сучасні підходи до модульного програмування.
8. Допомагає зрозуміти життєвий цикл розробки програмного забезпечення.
9. Розвиває системне мислення при аналізі та постановці задач.
10. Підкреслює важливість автоматизованого тестування.
11. Програма є багатофункціональною, вирішуючи три окремі, але взаємопов'язані задачі.
12. Код є модульним і добре структурованим.
13. Використано об'єктно-орієнтовані (або процедурні з модулями) принципи для кращої організації коду.
14. Реалізовано обробку винятків (або перевірку помилок) для надійнішої роботи з файлами.
15. Забезпечено коректну роботу з українськими символами (UTF-8, wchar\_t, локалі).
16. Точно дотримано вимог до транслітерації згідно з Постановою КМУ.
17. Функція s\_calculation інтегрована як приклад використання зовнішньої бібліотеки.
18. Реалізовано конвертацію чисел у двійковий формат, що показує розуміння бітових операцій.
19. Вихідний файл формується поступово, через дозапис, що ефективно для великих обсягів даних.
20. Використано стандартні бібліотеки C++ ( <fstream>, <string>, <chrono> тощо) для реалізації функціоналу.
21. Демонструє вміння працювати з файловим введенням/виведенням.
22. Підтверджує навички роботи з різними типами даних (char, wchar\_t, float, unsigned int).
23. Розвиває навички налагодження (debug) коду.
24. Поглиблює розуміння концепцій статичних бібліотек та їх інтеграції.
25. Надає досвід у написанні документації (аналіз, постановка, висновок).
26. Вміння створювати та керувати проектами в Code::Blocks IDE.
27. Навички роботи з Git (якщо ви використовували для версіонування, як зазначено у висновку).
28. Покращено аналітичні здібності при декомпозиції складної задачі на менші підзадачі.
29. Практичний досвід у написанні main функції як тестового драйвера.
30. Набуто досвіду у плануванні тестування та створенні тест-кейсів.
31. Код є читабельним і коментованим.
32. Дотримано принципів чистого коду (якщо це так).
33. Програма пройшла успішне автоматизоване тестування.
34. Результати тестування підтверджують коректність реалізації.
35. Програма демонструє стабільну роботу в різних сценаріях.
36. Обробка помилок (наприклад, неіснуючих файлів) підвищує надійність.
37. Використання std::locale підвищує сумісність з багатомовним контентом.
38. Забезпечено точність обчислень для числових даних.
39. Двійкове перетворення є точним для натуральних чисел.
40. Використання std::ios\_base::app для дозапису є ефективним та запобігає втраті даних.
41. Робота є комплексним завданням, що об'єднує кілька ключових тем курсу.
42. Продемонстровано здатність до самостійного вирішення поставлених завдань.
43. Показано відповідальність та уважність до деталей (наприклад, формат виведення).
44. Це завдання було цікавим та стимулюючим для навчання.
45. Отримані знання та навички будуть корисними для майбутніх проектів.
46. Вирішення задачі вимагало не тільки кодування, але й логічного мислення.
47. Розуміння обмежень (наприклад, спрощення транслітерації) і готовність їх обговорювати.
48. Готовність до подальшого вдосконалення та розширення функціоналу.
49. Дотримання встановлених дедлайнів та вимог до оформлення.
50. Робота відображає значний прогрес у вивченні С++ та принципів програмування.