ПЕРЕДМОВА

Важливою складовою навчального процесу є лабораторні заняття, в ході яких студенти у спеціально обладнаних лабораторіях та під керівництвом викладача виконують експериментально-дослідницьку роботу в межах конкретної дисципліни з метою набуття практичних навичок за обраною ними спеціальністю.

Якщо під час лекцій, як одного з найважливіших видів навчальних занять, студентам надаються основи наукових знань, формується науковий світогляд, розкриваються найбільш складні питання матеріалу дисципліни "Модульне програмування", то лабораторні заняття спрямовані на прищеплення практичних навичок, тобто реалізують один з головних принципів навчання – зв'язок Теорії з Практикою.

В забезпеченні означеного ключову роль відіграють організаційно-методична складова навчального процесу та відповідальне ставлення до нього студента. Відповідно до Положення про організацію освітнього процесу у Кіровоградському національному технічному університеті, лабораторне заняття включає проведення інструктажу з техніки безпеки, поточного **контролю підготовленості студента** до виконання конкретної лабораторної роботи, виконання завдань з теми заняття, оформлення індивідуального звіту з виконаної роботи та його захист перед науково-педагогічним працівником. Водночас, в межах самостійної роботи **у вільний від аудиторних занять час студент зобов'язаний**, зокрема, опрацьовувати навчальний матеріал дисципліни, літературні джерела та здійснювати належну підготовку до виконання лабораторних робіт.

Отже, слід **ретельно готуватись до кожного заняття**. Підготовка до чергової лабораторної роботи здійснюється студентом самостійно з обов'язковим опрацюванням навчальної, довідникової, наукової літератури задля ґрунтовного вивчення теоретичних положень дисципліни "Модульне програмування", винесених на лабораторну роботу, а також самоконтролю підготовленості до виконання завдань за темою заняття.

ЛАБОРАТОРНА РОБОТА № 4

Оброблення даних складових типів з файловим введенням/виведенням

Мета роботи — набути практичних навичок реалізації мовою програмування C++ у кросплатформовому середовищі Code::Blocks програмних модулів та засобів для оброблення даних типів масив, структура, об'єднання, множина, перелік з використанням файлових потоків.

Час виконання лабораторної роботи

4 академічні години.

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 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.).

Завдання до лабораторної роботи

1. Реалізувати програмні модулі розв'язування задач 4.1-4.3 як складових статичної бібліотеки *Прізвище*Lib, створеної під час виконання лабораторної роботи № 2.
2. Реалізувати тестовий драйвер перевірки програмних модулів розв'язування задач 4.1-4.3.

Порядок виконання лабораторної роботи та методичні вказівки

1. У викладача отримати умови задач 4.1-4.3 за варіантом.
2. До звіту з лабораторної роботи (далі — звіт) записати тему й мету роботи, номер варіанту, завдання.
3. Почергово здійснити аналіз задач 4.1, 4.2, 4.3.
4. Здійснити аналіз вимог до програмних модулів та вмісту вхідного текстового файлу (див. завдання за варіантом).
5. На основі результатів аналізу задач 4.1-4.3 і вимог розробити три тест-сьюти задля проведення Unit-тестування; належно їх задокументувати та включити до звіту;
   - Preliminary Steps – ім'я вхідного файлу та його вміст (текст) і/або ім'я вихідного файлу;
   - Action (test steps) – виклик відповідного модуля з конкретними вхідними даними;
   - Expected Result – вміст вхідного/вихідного файлу (текст);
6. Почергово здійснити функціональну декомпозицію задач 4.1-4.3, детальне проектування (алгоритмізацію); одержані артефакти задокументувати й включити до звіту;
   - вхідні дані модуля – ім'я вхідного і/або вихідного файлу;
   - якщо вихідний файл існує, то його вміст знищується, інакше – створюється;
7. В Code::Blocks IDE відкрити проект статичної бібліотеки *Прізвище*Lib (\MP*Прізвище*_Lab2\MyLib\), створеної під час виконання лабораторної роботи № 2.
8. За артефактами процесу детального проектування здійснити конструювання програмних модулів: мовою програмування C++ описати належно іменовані функції, які за інтерфейсом реалізовують розв'язування задач 4.1, 4.2 та 4.3 відповідно.
9. Скомпілювати проект статичної бібліотеки *Прізвище*Lib.
10. В Code::Blocks IDE відкрити проект заголовкового файлу *Прізвище*Module, створений під час виконання лабораторної роботи № 2 (\MP*Прізвище*_Lab2\MyLib\), та додати прототипи

реалізованих у статичній бібліотеці *Прізвище*Lib функцій розв'язування задач 4.1, 4.2, 4.3.

11. В Code::Blocks IDE створити проект консольного додатка й мовою програмування C/C++ реалізувати тестовий драйвер для Unit-тестування функцій задач 4.1, 4.2 й 4.3 з *Прізвище*Module.h за допомогою розроблених тест-сьютів та вхідного і/або вихідного текстового файлу.
    - ім'я файлу/файлів — аргумент(и) функцій з *Прізвище*Module.h;
    - контрольні приклади рекомендовано описати константними масивами, елементи яких у циклі передаються на оброблення відповідною функцією з *Прізвище*Module.h;
    - є можливим забезпечення тестовим драйвером або повністю автоматизованого, або напівавтоматизованого Unit-тестування:
      - *для автоматизованого* слід реалізувати алгоритм виконання відповідного тест-кейса: *A1)* драйвер за Preliminary Steps створює вхідний файл та записує в нього вхідний текст і/або створює вихідний файл, *A2)* за Action викликається функція з аргументами – ім'я/іменами файлів, *A3)* відкривається модифікований функцією з кроку *A2* файл, зчитується текст з нього та порівнюється з текстом із поля Expected Result; *A4)* результат порівняння (Test Result passed / failed) виводиться у стандартний або файловий потік тестувальнику;
      - *для напіватоматизованого* кроки *A1*, *A3* та *A4* алгоритма автоматизованого виконання тест-кейса реалізовуються тестувальником вручну за допомогою текстового редактора, а крок *A2* — тестовим драйвером.
    - у випадку невиконання тест-кейса(ів) слід здійснити відлагодження відповідної функції, перекомпілювати проект статичної бібліотеки й процес тестування повторити.
12. За допомогою розробленого тест-драйвера здійснити тестування функцій розв'язування задач 4.1-4-3 з *Прізвище*Module.h.
13. Результати модульного тестування належно задокументувати й включити до звіту.
14. Розроблений тестовий драйвер (exe-файл) скопіювати у \MP*Прізвище*_Lab4\my_software\.

15. Затвердити у викладача реалізоване під час виконання лабораторної роботи програмне забезпечення.
16. Лістинги розроблених функцій статичної бібліотеки *Прізвище*Lib та тестового драйвера, текст заголовкового файлу *Прізвище*Module.h включити до звіту.
17. Одержані результати виконання завдань зберегти на носій (флеш-накопичувач, хмарне сховище даних тощо) з метою забезпечення можливості їх подальшого використання під час виконання наступних лабораторних робіт.
18. Здійснити аналіз процесу виконання лабораторної роботи й одержаних результатів, сформулювати обґрунтовані висновки (підсумки)[1] обсягом не менше ½ сторінки машинописного тексту та включити їх до звіту; у висновках варто також окремо зазначити особисті враження від процесу виконання завдань, викласти вмотивовані пропозиції, обґрунтовані рекомендації, зауваження, конструктивну критику[2] тощо.
19. Підготувати звіт з лабораторної роботи відповідно до встановлених вимог до його структури, змісту й оформлення.
20. Подати викладачу звіт до захисту.

Контрольні запитання і завдання

1. Яке призначення і синтаксис запису блоку-контроля `try - throw - catch` у мові програмування C/C++?
2. Наведіть приклад опису й використання міжмодульної змінної.
3. Яку область видимості матимуть об'єкти (змінні, типи тощо), описані в тілі функції `main()`?
4. Здійсніть порівняльний аналіз змінної типу `enum` та масиву.

[1] *висновки*, як результат розумової діяльності студента, повинні, зокрема, містити стисле викладення здобутих в процесі виконання лабораторної роботи результатів, реалізованих ідей, опис проблем, які виникали під час реалізації завдань, та шляхи їх вирішення; структура підсумків має бути логічною і охоплювати весь процес виконання лабораторної роботи тощо.

[2] *критика* є розглядом і оцінкою когось чи чогось з метою виявлення й усунення вад, хиб; під *конструктивною* слід розуміти критику, після якої стає зрозумілим, як саме виправити помилку й не допускати її в майбутньому.

5. Дайте визначення потоку й файлового потоку. Чим файловий потік відрізняється від стандартного?
6. Запропонуйте універсальний алгоритм реалізації мовою програмування C++ процесу читання/запису даних з/у файл за допомогою файлових потоків.
7. Який об'єм (у байтах) матиме текстовий файл, якщо в нього записано значення фундаментальної математичної константи π з точністю 10^6 десяткових цифр після коми?
8. Яким чином у ПЗ можливо реалізувати перевірку статуса відкриття файлового потоку (відкрито/не відкрито)?
9. Чим текстовий потік відрізняється від двійкового з погляду реалізації процесу оброблення даних з них?
10. Які розрізняють режими відкриття файлових потоків у C++?
11. Класи яких файлових потоків реалізовано у `fstream`? Мовою програмування C++ наведіть приклади створення файлових об'єктів та відкриття відповідних потоків.
12. Перелічіть описані в `ios` C++ константи режимів відкриття файлових потоків та призначення кожної з них.
13. За допомогою яких функцій-членів файлових об'єктів реалізовується відкриття й закриття потоку, визначення кінця файлу?
14. Запропонуйте алгоритм видалення заданої послідовності символів (наприклад, слова) з текстового файла.
15. Перелічіть складові типи даних C/C++ та сформулюйте особливості оголошення й оброблення їх об'єктів (змінних) порівняно з об'єктами простих типів.
16. Яким чином у C/C++ здійснюється перевірка наявності елемента у множині, його додавання, вилучення?
17. Як у C/C++ здійснюється явне й неявне перетворення типів даних?
18. Перелічіть випадки (задачі), у яких є доцільним використання змінних типу `union`. Наведіть приклади їх реалізації мовою C++.
19. Запропонуйте алгоритм програмного модуля перевірки наявності у текстовому файлі заданого натурального числа.
20. Що міститиме змінна `symbol_transaction` при:
    `short symbol_transaction = sizeof short('R');`
    Відповідь поясніть та доведіть експериментально.