

МОДУЛЬ 3

Лабораторна робота 3.1

ДОСЛІДЖЕННЯ РОБОТИ З ФАЙЛАМИ ТА ПРОЕКТУВАННЯ ЗВ'ЯЗКІВ МІЖ СУТНОСТЯМИ

Мета роботи – дослідити роботу з файлами з використанням потоків у мові C# при роботі з різними пов'язаними сутностями.

Завдання

1. Дослідити структуру класів потокового введення/виведення для роботи з файлами у мові C#.
2. Спроектувати, представити у вигляді діаграми класів функціонал згідно з варіантом.
3. Розробити програму на мові C#, яка відповідає вимогам у завданні та варіанті. Для демонстрації роботи використати текстові повідомлення на консолі через операції вводу-виводу. Програма повинна показати застосування класів потокового введення/ виведення для роботи з файлами шляхом запису, читання та маніпуляцій над даними спроектованих сутностей та зв'язків між ними.
4. Діаграма(-и) та вихідний код повинні відповідати базовим принципам проектування: ООП, composition over inheritance, loose coupling – high cohesion, inversion of control (IoC).
5. Для отримання балів, що відповідають «задовільно» необхідно реалізувати завдання, відповідно варіанту. Операції роботи із файлами, вводу-виводу повинні бути в окремих класах, а не в класах бізне-сущностей.
6. Для отримання балів, що відповідають «добре» повинно бути реалізовано усе, що на «задовільно». А також:
 - a. В окремий проект (project) рішення (solution) виділити операції читання-запису у файл. Тут не повинно бути ніяких маніпуляцій з даними чи ввід-вивід на консоль. Тобто в рішенні буде мінімум 2 проекти
 - b. Операції вводу-виводу на екран – в окремому класі **ConsoleMenu**.
 - c. Також необхідно використати абстракції (абстрактні класи та інтерфейси), для демонстрації зв'язків між сутностями у варіанті.
7. Для отримання балів, що відповідають «відмінно», спроектувати зв'язки між сутностями таким чином, щоб можна було легко додавати інші пов'язані типи (наприклад, Pupil чи Musician) без необхідності змінювати існуючі. А також – нову поведінку. Наприклад, коли операцію Play() може виконувати музикант та студент.
8. *Завдання підвищеної складності – працювати з одним єдиним файлом-джерелом даних, записуючи та читаючи дані різних типів та сутностей.

Методичні рекомендації

При виконанні роботи слід пам'ятати:

- різні характеристики/ сутності та специфічні операції не повинні бути в одному класі/ модулі;
- функціонал повинен бути легко розширюваним без необхідності внесення змін в існуючий код (чи при мінімальних змінах) – якщо в завданні не сказано протилежне;
- при необхідності реалізації однотипних операцій а також для зменшення зв'язку між класами та надання більшої гнучкості при розширенні/ модифікації спроектованої програми варто використати абстракцію;
- код не повинен повторюватися і бути читабельним;
- повинно бути реалізовано тільки те, що вказане в завданні і ніякого зайвого функціоналу;
- в кожному проекті можуть бути також допоміжні класи та інтерфейси. Так само, як і в кожному класі можуть бути також інші члени, окрім тих, що вказані в завданні;
- інтерфейси та абстрактні класи в C# мають схожі характеристики, але варто обирати те чи інше рішення, залежно від особливостей задач – для одних краще підходить інтерфейс, для інших – абстрактний клас.

Вимоги по реалізації застосування

Використовувати багаторівневу архітектуру в даній лабораторній не є необхідним (але і не забороняється).

Головним критерієм якості є реалізація функціоналу та відповідність базовим принципам проектування. Для побудови залежностей між класами та їх об'єктами використовувати залежності, асоціації (в тому числі композиції та агрегації), наслідування (узагальнення) та реалізації.

За необхідності можна додавати нові властивості, сутності та розширювати функціонал. Але ця необхідність повинна бути обґрунтована.

Управління та відповідь застосування здійснюється через консоль. Використання консолі, наприклад, для виведення повідомлень, потрібно внести за межі функціоналу предметної області. Тобто, вивід повідомлень на консоль повинен бути за межами методів, що виконують певні дії. Наприклад, вивід повідомлення про те, що студент навчається на 3-му курсі повинне бути окремо від методу, який визначає, що він навчається саме в групі 3-го курсу.

Методи класів предметної області мають змінювати певним чином стан об'єкту, а не просто виводити повідомлення на консоль (наприклад, переведення студента на наступний курс навчання).

Для демонстрації роботи програми необхідно підготувати набір вхідних даних.

Зберігання даних у файлі

Дані у файлі повинні зберігатися в наступному форматі:

ТипДаних1 НазваОб'єкту1

{ «атрибут1»: «значенняАтрибуту1»,
 «атрибут2»: «значенняАтрибуту2»};

ТипДаних2 НазваОб'єкту2

{ «атрибут1»: «значенняАтрибуту1»,
 «атрибут2»: «значенняАтрибуту2»};

Наприклад:

Student VasiaPupkin

{ “firstname”: “Vasia”,
“lastname”: “Pupkin”,
“studentId”: “KB123456”};

Teacher IvanIvanov

{ “firstname”: “Ivan”,
“lastname”: “Ivanov”};

Результатом читання інформації з файлу буде масив (-и) одного чи кількох типів, залежно від типів, вказаних у файлі. Наприклад, результатом читання фрагменту з попереднього прикладу буде 1 об'єкт типу Student з відповідними значеннями 3-х атрибутів та 1 об'єкт типу Teacher з відповідними значеннями 2-х атрибутів.

Допускається використання кількох файлів: в окремий файл будуть зберігатися дані одного окремого типу.

Використання колекцій **забороняється**.

Проектування сутностей

Передбачити роботу з класом Student та ще двома іншими (за варіантом в табл. 1). Обов'язкові атрибути кожного з них: FirstName, LastName. Додати атрибути до класів по доцільності. Також визначити їх обов'язковість.

Обов'язковість – значить, що для об'єкту заданого типу обов'язково потрібно задати значення для даного атрибуту. Значення необов'язкового атрибуту може бути задане за умовчанням.

Наприклад, в класі Student варто додати ще атрибут StudentID (студентський квиток). Більш того – він є обов'язковим, оскільки студент не може навчатися в університеті без студентського квитка.

Також передбачити виконання специфічних для кожної сутності дій: Study() – для студента дія передбачає навчання у ВУЗі, а для інших типів – самонавчання (за необхідності), Teach() – дія специфічна для людини, що працює викладачем/ тренером, Drive() – дія специфічна для людини з водійським посвідченням.

За доцільністю варто додати інші дії аналогічно атрибутам.

Для класу Student необхідні атрибути та додаткові операції визначені в таблиці з варіантами.

Робота з даними

Створити базу даних (БД) про студентів та додаткових сутностей (за варіантом), яка зберігається у файлі (-ах). Для чого необхідно описати операції введення даних про N осіб (різних сутностей) у БД ($N > 5$), отримання даних про осіб із БД та видалення цих даних. А також виконати пошук особи(-ів) за прізвищем та унікальним ідентифікатором, обробити базу даних згідно завдання за варіантом.

Зберігати та читувати дані з файлу, вводити та виводити на консоль потрібно поелементно (а не об'єктом).

Для перевірки коректності введеної інформації використати **регулярні вирази**, як мінімум для: студентський квиток, залікова книжка, ідентифікаційний код, паспорт, військовий квиток мають визначений формат. В імені, прізвищі, країні, курсі перевіряти введення допустимих символів.

№	Елементи класу Student	Варіанти		
		Операції зі студентами	Додаткові сутності	Додаткові вміння
1	Прізвище, Ім'я, Курс, Студентський квиток, Дата народження у форматі XX-XX-XXXX	Обчислити кількість студентів 3-го курсу, які народилися влітку. Отримати їх дані з файлу	Teacher, Astronaut	Співати
2	Прізвище, Ім'я, Курс, Студентський квиток, Місто прибуття, серія та номер паспорту	Обчислити відсоток студентів 1-го курсу, які приїхали з інших міст. Отримати їх дані з файлу	TaxiDriver, Acrobat	Танцювати
3	Прізвище, Ім'я, Курс, Студентський квиток, Стать, Місце проживання, для тих, хто проживає в гуртожитку – формат «номер гуртожитку.кімната»	Обчислити кількість студенток 1-го курсу, які проживають у тургожитку. Отримати їх дані з файлу	Musician, Pilot	Кататися на ковзанах
4	Прізвище, Ім'я, Курс, Студентський квиток, Служба в армії, для тих, хто служив в армії – номер військового квитка	Обчислити кількість студентів 5-го курсу, які служили в армії. Отримати їх дані з файлу	FootballPlayer, Lawyer	Декламувати вірші
5	Прізвище, Курс, Студентський квиток, Середній бал, Країна, номер залікової книжки	Обчислити кількість студентів 3-го курсу, які проживають в Україні. Отримати їх дані з файлу	McDonaldsWorker, Manager	Грати в шахи
6	Прізвище, Ім'я, Курс, Студентський квиток, Хобі - спорт, ідентифікаційний код	Обчислити кількість студентів 2-го курсу, які займаються спортом. Отримати їх дані з файлу	Courier, Fireman	Грати на гітарі
7	Прізвище, Ім'я, Ріст, Вага, Студентський квиток, серія та номер паспорту	Обчислити кількість студентів з ідеальною вагою тіла (Ріст – 110 = Вага) . Отримати їх дані з файлу	Librarian, SoftwareDeveloper	Кататися на велосипеді
8	Прізвище, Ім'я, Курс, Студентський квиток, Дата народження у форматі XX.XX.XXXX	Обчислити кількість студентів 4-го курсу, які народилися навесні. Отримати їх дані з файлу	Baker, Entrepreneur	Стрибати з парашутом
9	Прізвище, Ім'я, Курс, Студентський квиток, Дата	Обчислити кількість студентів 2-го курсу, які народилися	Painter, Farmer	Плавати

	народження у форматі XXXX XX XX	взимку. Отримати їх дані з файлу		
10	Прізвище, Ім'я, Курс, Студентський квиток, Стать, Місце проживання, номер залікової книжки	Обчислити кількість студенток 5-го курсу, які постійно проживають у Києві. Отримати їх дані з файлу	Joiner, Photographer	Множити велики числа
11	Прізвище, Ім'я, Курс, Студентський квиток, Стать, Місце проживання, для тих, хто проживає в гуртожитку – формат «номер гуртожитку-кімнати»	Обчислити кількість студентів чоловіків 3-го курсу, які проживають у гуртожитку. Отримати їх дані з файлу	Seller, Gardener	Спати стоячи
12	Прізвище, Курс, Студентський квиток, Середній бал, Країна, номер закордонного паспорту	Обчислити кількість іноземних студентів-відмінників 1-го курсу. Отримати їх дані з файлу	Tailor, Singer	Пірнати
13	Прізвище, Ім'я, Курс, Студентський квиток, Стать, Середній бал, ідентифікаційний код	Обчислити кількість студенток-відмінниць 5-го курсу. Отримати їх дані з файлу	Storyteller, Dentist	Готувати
14	Прізвище, Ім'я, Курс, Студентський квиток, Стипендія, серія та номер паспорту	Обчислити кількість студентів 4-го курсу, які не отримують стипендії. Отримати їх дані з файлу	Postman, Mountaineer	Малювати
15	Прізвище, Ім'я, Курс, Студентський квиток, Стать, Середній бал, номер залікової книжки	Обчислити кількість студентів-чоловіків 2-го курсу, які займаються на відмінно. Отримати їх дані з файлу	Doctor, Mechanic	Вигадувати історії

Контрольні запитання

- Що таке потік?
- Наведіть приклади режимів відкриття файлів.
- Поясніть різницю між файлами з послідовним та довільним доступом.
- Поясніть, як реалізується довільний доступ у файлах.
- Опишіть ієархію класів потокового введення/виведення в C#.
- Наведіть методи читання/запису для потоку FileStream в C#.
- Поясніть, як створити символний потік в C#.
- Поясніть, як виконується перенаправлення потоків в C#.
- Наведіть приклади методів для читання/запису значень наперед визначених типів у двійкові потоки в C#.
- Посніть використання регулярних виразів.
- В чому відмінність між композицією та наслідуванням. Який зв'язок і коли варто використовувати? Чому?
- Поясніть твердження «composition over inheritance». Чому воно виникло?
- В чому різниця між абстрактним класом та інтерфейсом?
- В чому полягає користь використання абстракцій замість конкретних реалізацій? Наведіть приклади.
- В чому полягає принцип інверсії контролю (inversion of control – IoC)?