Завдання для курсових робіт з дисципліни «Об'єктно-орієнтоване програмування»

1. В якості типу застосування обрати консольне застосування з командним рядком, як виняток, win forms, wpf.
2. Спроектувати та реалізувати систему класів, в основу якої покладено логічну структуру даних, наведену у варіанті, для накопичення та обробки даних домену відповідно варіанту курсової роботи.
3. Структура програмної системи курсового проекту:
   Код програмної системи має складатись не менш як з трьох частин (окремих проектів) відповідно до багатошарової архітектури системи, де шарами архітектури є: шар (рівень) доступу до даних (DAL), шар бізнес-логіки (BLL), шар представлення (інтерфейс програмної системи) (PL). Тип програмного модулю – DLL.
   Шар доступу до даних організувати таким чином, щоб він забезпечував збереження даних у файлах за допомогою серіалізації або у БД із застосуванням Entity Framework. Тип програмного модулю – DLL.
   В шарі бізнес-логіки побудувати класи, що представляють об'єкти та дії над ними відповідно до предметної області, наприклад, читачі, книги та абонементи, читач може взяти книгу на свій абонемент при її наявності у сховищі та повернути її у заданий термін. Для створення наборів об'єктів предметної області (студенти, викладачі, книги, замовлення, страви, тощо) використовувати класи-узагальнені колекції. Всі операції бізнес-логіки використовують об'єктну модель даних. Якщо дані потрібно отримати зі сховища даних або зберегти у сховищі, шар бізнес-логіки звертається до шару доступу до даних, передаючи\отримуючи збережені дані.
   Шар представлення – це додаток типу відповідного до обраного інтерфейсу (консольний чи GUI). Людина-користувач через шар представлення взаємодіє з програмною системою через інтерфейс, реалізований шаром представлення. Для виконання дій на вимогу користувача шар представлення звертається до шару бізнес-логіки, передаючи в нього ведені користувачем дані та команди виконати певні дії. Результати виконаних дій, отримані від шару бізнес-логіки, шар представлення відображує користувачу у консолі\формах.
   Принцип багатошаровості при будуванні архітектури є обов'язковим до застосування!
4. Реалізувати обов'язковий контроль коректності введених даних в шарі представлення.
5. Реалізувати перевірку виняткових ситуацій в шарі бізнес-логіки та доступу до даних; в тому числі – при роботі з даними. При необхідності створити власні класи виключень (наприклад, виключення для ситуації перевищення ліміту кількості книжок на

абонементі в бібліотеці). Генерація/ виникнення вийняткових ситуацій, як правило відбувається на іншому рівні (рівнях), ніж їх обробка.

6. Написати модульні тести, використовуючи певний фреймворк (MSTest, Nunit, Xunit, тощо), до бізнес логіки. Мінімальне покриття тестами – 100% функціоналу однієї з сутностей, а також – мінімум 50% функціоналу усіх іншхи сутностей. Покриття можна продемонструвати відповідними засобами, наприклад AxoCover, CodeCoverage та ін (допускається також детальне описання функціоналу, покритого модульними тестами). Модульні тести повинні бути окремим проектом в рішенні. Для оформлення коду модульних тестів овоб’язково застосовувати принцип Triple A.
7. Сутності за зв’язки між ними повинні бути спроектовані, відповідно до базових правил ООП, composition over inheritance, loose coupling – high cohesion, inversion of control (IoC). А також - з використанням узагальнень.
8. При написанні коду застосувати правила «хорошого» стилю програмування. А саме: створювати мнемонічні ідентифікатори, чітко форматувати код, застосувати при необхідності коментарі. Не використовувати public полів у класах. За необхідності доступу до них використати властивості, індексатори чи окремі методи-аксесори. Опис класів наводити в окремих файлах. За неохайне оформлення коду можливе зниження оцінки за курсову роботу!
9. Пояснювальна записка має містити наступні складові:
    a. Титульний лист
    b. Зміст
    c. Опис завдання
    d. Опис шарів проекту та загальна архітектура проекту
    e. Опис інших компонентів проекту (при наявності)
    f. Діаграма класів з усіми зв’язками та поясненнями
    g. Описання деяких особливостей реалізації (допускається приведення прикладів з кількома рядками коду)
    h. Описання функціоналу та використання (допускається використання кількох скрішнотів)
    i. Використані джерела
    j. Лістінг коду проекту (**не друкувати, а викласти в репозиторій/ додати архівом**)
    k. Скріншоти (результати введення\виведення) проекту (**не друкувати, а а викласти в репозиторій/ додати архівом**)
10. До розробленого застосування та оформленої пояснювальної записки розробити презентацію та доповідь (приблизно 5 хв.).