Лабораторна робота № 5

Звіт

**Семенченко Олександр КМ-33**

1. Опишіть різницю між шаблонами Проксі та Адаптер у Java. Наведіть приклад сценарію, де кожен з них буде найбільш доцільним.

**Проксі це посередник**, який контролює доступ до об'єкта (наприклад, для кешування, безпеки або лінійної ініціалізації). **Приклад:** завантаження зображення лише під час його відображення. **Адаптер** конвертує інтерфейс одного класу в інший, щоб зробити об'єкти сумісними. **Приклад:** використання нового API в старій системі.

1. Як реалізувати шаблон Декоратор для створення системи знижок у додатку електронної комерції?

Декоратор дозволяє динамічно додавати функціональність до об'єктів. У системі знижок він може бути використаний для обчислення різних типів знижок.

1. Порівняйте шаблони Спостерігач (Observer) та Медіатор. У яких випадках один шаблон є кращим за інший при побудові слабко зв'язаних компонентів системи?

**Спостерігач організовує** одностороннє сповіщення між об'єктами: один об'єкт (суб’єкт) повідомляє багатьом спостерігачам про зміну стану. **Медіатор координує** взаємодію між багатьма об'єктами, усуваючи прямі зв'язки між ними. Спостерігач: якщо зміни одного об'єкта мають сповіщати інші компоненти (просте сповіщення). Медіатор: якщо необхідно управляти складними взаємодіями між багатьма компонентами.

1. Запропонуйте приклад використання шаблону Ланцюг відповідальності для реалізації системи обробки запитів у корпоративній мережі, де кожен обробник має власні правила фільтрації.

Шаблон дозволяє передавати запити по ланцюгу обробників, поки їх не обробить відповідний. Приклад: система обробки запитів у корпоративній мережі.

1. Як шаблон Стан може бути використаний для моделювання життєвого циклу замовлення в системі управління замовленнями? Наведіть структуру класів та опишіть переходи між станами.

Інтерфейс OrderState, він містить методи для дій (наприклад, next(), cancel()).

Класи станів: NewOrder, Processing, Shipped, Delivered, Canceled.

Клас Order, він містить поточний стан та делегує методи.

Переходи:

NewOrder -> Processing -> Shipped -> Delivered.

Будь-який стан -> Canceled.

1. Порівняйте шаблони Команда та Стратегія. Наведіть приклад, коли кожен з них буде найбільш ефективним при проектуванні системи.

Команда інкапсулює дію у вигляді об'єкта для її відкладеного виконання, скасування або збереження. Стратегія інкапсулює алгоритми, дозволяючи змінювати їх під час виконання.  
Команда: для організації дій як об'єктів (збереження історії операцій).

Стратегія: для вибору поведінки або алгоритму під час виконання.

1. Як реалізувати Композит для створення ієрархічної структури файлової системи з підтримкою операцій над окремими файлами та директоріями?

Інтерфейс FileSystemComponent: описує спільні операції, наприклад, getSize(), display(). Клас File яка реалізує операції для файлу. Клас Directory, він зберігає список компонентів і викликає їхні методи рекурсивно.

1. Опишіть механізм реалізації шаблону Міст для абстракції графічних елементів інтерфейсу, де можна незалежно змінювати вигляд та поведінку елементів.

Інтерфейс Renderer визначає реалізацію вигляду елементів (наприклад, renderCircle(), renderRectangle()). Конкретні реалізації VectorRenderer та RasterRenderer які реалізують методи відображення у векторному або растровому вигляді. Абстракція Shape яка описує графічний елемент із посиланням на Renderer. Конкретні абстракції (Circle, Rectangle) які реалізують специфічну поведінку.

1. Опишіть використання Шаблонного методу для створення фреймворку для генерації звітів з різними налаштуваннями форматування.

Абстрактний клас Report який містить шаблонний метод generateReport(), який визначає порядок кроків (наприклад, fetchData(), formatData(), exportReport()).Конкретні підкласи (PDFReport, HTMLReport) які реалізують конкретні кроки форматування або експорту.

1. Як за допомогою шаблону Відвідувач можна реалізувати систему обчислення сумарної вартості елементів у складній ієрархічній структурі (наприклад, склад комп'ютерних компонентів)?

Інтерфейс Component який описує метод accept(Visitor visitor), який дозволяє відвідувати елементи. Конкретні елементи (CPU, RAM, HardDrive) які реалізують метод accept(), передаючи себе до відвідувача. Інтерфейс Visitor який описує операції, які можуть бути виконані над елементами (наприклад, visitCPU(), visitRAM()). Конкретний відвідувач (CostCalculator)який реалізує методи обчислення вартості для кожного типу компонента.