**Методичні вказівки до лабораторних робіт з курсу**

**«Інженерія програмного забезпечення»**

**Лабораторна робота №6**

**Тема**: Шаблони поведінки - 2. Шаблони Strategy, Chain of Responsibility та Visitor.

**Мета**: Вивчення шаблонів поведінки. Отримання базових навичок з застосування шаблонів Strategy, Chain of Responsibility та Visitor.

**Завдання**

1. Повторити шаблони поведінки для проектування ПЗ. Знати загальну характеристику шаблонів поведінки та призначення кожного з них.

2. Детально вивчити шаблони поведінки для проектування ПЗ — Strategy, Chain of Responsibility та Visitor. Для кожного з них:

* вивчити Шаблон, його призначення, альтернативні назви, мотивацію, випадки коли його застосування є доцільним та результати такого застосування;
* знати особливості реалізації Шаблону, споріднені шаблони, відомі випадки його застосування в програмних додатках;
* вільно володіти структурою Шаблону, призначенням його класів та відносинами між ними;
* вміти розпізнавати Шаблон в UML діаграмі класів та будувати сирцеві коди Java-класів, що реалізують шаблон.

3. В підготованому проекті (ЛР1) створити програмний пакет com.lab111.labwork6. В пакеті розробити інтерфейси і класи, що реалізують завдання (згідно варіанту) з застосуванням одного чи декількох шаблонів (п.2). В розроблюваних класах повністю реалізувати методи, пов'язані з функціюванням Шаблону. Методи, що реалізують бізнес-логіку закрити заглушками з виводом на консоль інформації про викликаний метод та його аргументи. Приклад реалізації бізнес-методу:

void draw(int x, int y){

System.out.println(“Метод draw з параметрами x=”+x+” y=”+y);

}

4. За допомогою автоматизованих засобів виконати повне документування розроблених класів (також методів і полів), при цьому документація має в достатній мірі висвітлювати роль певного класу в загальній структурі Шаблону та особливості конкретної реалізації.

Варіанти (№зк **mod** 10)

1. Визначити специфікації класу, який містить масив цілих чисел та метод його сортування. Забезпечити можливість динамічної зміни алгоритму та напрямку сортування шляхом зовнішньої параметризації.
2. Визначити специфікації класу, який містить таблицю та метод її відображення у вигляді діаграми. Забезпечити можливість динамічної зміни типу діаграми шляхом зовнішньої параметризації.
3. Визначити специфікації класу, який містить математичну функцію та метод її відображення у вигляді графіка. Забезпечити можливість динамічної зміни системи координат графіка (декартова, полярна тощо) шляхом зовнішньої параметризації.
4. Визначити специфікації класів, що реалізують контейнери для цілих чисел та текстових строк з можливістю їх сортування. Забезпечити можливість динамічної зміни алгоритму сортування шляхом зовнішньої параметризації. Реалізація алгоритму сортування має бути незалежною від типу даних, що сортуються.
5. Визначити специфікації класів, що реалізують елементи графічного інтерфейсу користувача — панелі (композит) та кнопки (компонент). Реалізувати децентралізований механізм обробки події переміщення курсору миші. Кількість компонентів інтерфейсу, які реагують на цю подію, може змінюватись динамічно.
6. Визначити специфікації класів, що реалізують обробку HTTP-запитів різних типів (наприклад GET та POST). Реалізувати можливість динамічної зміни кількості обробників. Забезпечити децентралізацію та слабку зв'язаність обробників.
7. Визначити специфікації класів для елементу ігрового поля (комірки) та самого простору. Забезпечити слабку зв'язаність елементів. Реалізувати децентралізований механізм сумісної зміни стану елементів.
8. Визначити специфікації класів, що реалізують елементи графічного інтерфейсу користувача — панелі (композит) та кнопки (компонент). Реалізувати механізм додаткових операцій над структурою графічного інтерфейсу без зміни її елементів. В якості ілюстрації такого механізму розробити операцію підрахунку кількості елементів одного типу.
9. Визначити специфікації класів, що реалізують елементи структури комп'ютера (процесор, пам'ять, відеокарта тощо). Реалізувати механізм додаткових операцій над структурою комп'ютера без зміни її елементів. В якості ілюстрації такого механізму розробити операцію визначення потужності, що потребляє комп'ютер.
10. Визначити специфікації класів, що реалізують елементи мережевої структури (кабель, сервер, робоча станція). Реалізувати механізм додаткових операцій над мережевою структурою без зміни її елементів. В якості ілюстрації такого механізму розробити операцію визначення кошторису такої структури.

**Протокол**

Протокол має містити титульну сторінку (з номером залікової книжки), завдання, роздруківку діаграми класів, розроблений сирцевий код та згенеровану документацію в форматі JavaDoc.

**Матеріали**

Підготовка до лабораторної роботи здійснюється за допомогою книги:

*Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес* Приемы объектно-ориентированного проектирования. Паттерны проектирования = Design Patterns: Elements of Reusable Object-Oriented Software. — СПб: [«Питер»](http://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D1%82%D0%B5%D1%80_%28%D0%B8%D0%B7%D0%B4%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D1%82%D0%B2%D0%BE%29), 2007. — С. 366. — [ISBN 978-5-469-01136-1](http://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:BookSources/9785469011361) (также [ISBN 5-272-00355-1](http://ru.wikipedia.org/wiki/%D0%A1%D0%BB%D1%83%D0%B6%D0%B5%D0%B1%D0%BD%D0%B0%D1%8F:BookSources/5272003551))

За необхідності додаткової інформації можливо використання матеріалів з мережі Інтернет, наприклад:

Шаблони проектування програмного забезпечення

* + [Шаблони проектування програмного забезпечення](http://uk.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F)
  + [Шаблон проектирования](http://ru.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD_%D0%BF%D1%80%D0%BE%D0%B5%D0%BA%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)
  + [Обзор паттернов проектирования](http://citforum.ru/SE/project/pattern/)
  + [Объектно-ориентированное проектирование, паттерны проектирования (Шаблоны)](http://www.javenue.info/themes/ood/)
  + [David Gallardo. Шаблоны проектирования Java](http://khpi-iip.mipk.kharkiv.edu/library/extent/prog/jdp101/index.html)
  + [Design pattern (computer science)](http://en.wikipedia.org/wiki/Design_pattern_%28computer_science%29)
  + [Подготовка к собеседованию по Java/J2EE](http://cloud-cuckoo.com.ua/category/java/podgotovka-k-sobesedovaniyu-po-java/j2ee)

Шаблони поведінки

* [Шаблони поведінки](http://uk.wikipedia.org/wiki/%D0%A8%D0%B0%D0%B1%D0%BB%D0%BE%D0%BD%D0%B8_%D0%BF%D0%BE%D0%B2%D0%B5%D0%B4%D1%96%D0%BD%D0%BA%D0%B8)
* [Behavioral pattern](http://en.wikipedia.org/wiki/Behavioral_pattern)
* [Шаблоны поведения](http://khpi-iip.mipk.kharkiv.edu/library/extent/prog/jdp101/part6.html)