****

Міністерство освіти і науки України

Національний технічний університет України

“Київський політехнічний інститут імені Ігоря Сікорського”

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

Лабораторна робота №6  
ШАБЛОНИ «Abstract Factory»,

«Factory Method», «Memento»,

«Observer», «Decorator»

Виконав Перевірив:

студент групи ІА – 13: Мягкий М. Ю

Мельничук Дмитро

**Завдання:**

1. Ознайомитися з короткими теоретичними відомостями.

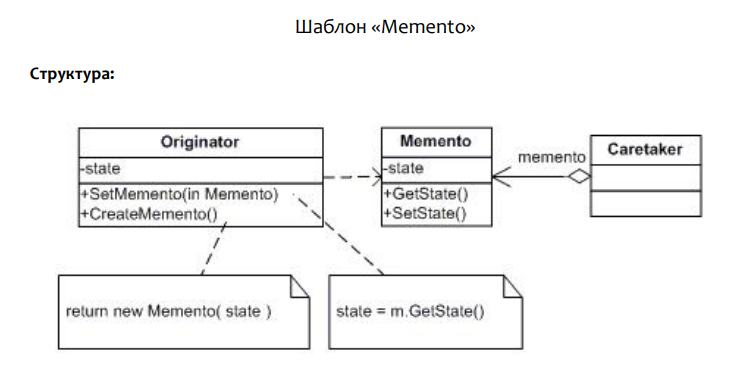
2. Реалізувати частину функціоналу робочої програми у вигляді класів та їх взаємодій для

досягнення конкретних функціональних можливостей.

3. Застосування одного з розглянутих шаблонів при реалізації програми.

**Хід роботи**

Шаблон використовується для збереження і відновлення стану об'єктів без порушення інкапсуляції. Об'єкт "мементо" служить виключно для збереження змін над початковим об'єктом (Originator). Лише початковий об'єкт має можливість зберігати і отримувати стан об'єкту "мементо" для власних цілей, цей об'єкт є "порожнім" для кого-небудь ще. Об'єкт Caretaker використовується для передачі і зберігання мементо об'єктів в системі. Таким чином вдається досягти наступних цілей: - Зберігання стану повністю відділяється від початкових об'єктів, що полегшує їх реалізацію; - Передача об'єктів мементо лягає на плечі Caretaker об'єктів, що дозволяє гнучкіше управляти станами об'єктів і спростити дизайн класів початкових об'єктів; - Збереження і відновлення стану реалізовані у вигляді двох простих методів і є закритими для кого-небудь ще окрім початкових об'єктів, таким чином не порушуючи інкапсуляцію. Шаблон "мементо" дуже зручно використати разом з шаблоном "команда" для реалізації "скасовних" дій - дані про дію зберігаються в мементо, а команда має можливість вважати і відновити початкове положення відповідних об'єктів.



// Клас Memento для зберігання стану чату

public class ChatMemento {

private final String chatState;

public ChatMemento(String state) {

this.chatState = state;

}

public String getChatState() {

return chatState;

}

}

// Клас Originator, який зберігає та відновлює стан чату

public class ChatOriginator {

private String chatState;

public void setChatState(String state) {

this.chatState = state;

}

public ChatMemento save() {

return new ChatMemento(chatState);

}

public void restore(ChatMemento memento) {

chatState = memento.getChatState();

}

// Додаткові методи та функціонал для роботи з чатом

}

// Клас Caretaker, який зберігає та відновлює стан чату

public class ChatCaretaker {

private ChatMemento memento;

public ChatMemento getMemento() {

return memento;

}

public void setMemento(ChatMemento memento) {

this.memento = memento;

}

}

public class Main {

public static void main(String[] args) {

// Створення комунікатора та чату

ChatOriginator chat = new ChatOriginator();

ChatCaretaker caretaker = new ChatCaretaker();

// Імітація взаємодії користувача з чатом та збереженням стану

chat.setChatState("Привіт, як справи?");

caretaker.setMemento(chat.save());

// Імітація подій чату, зміна стану

chat.setChatState("Дуже добре! Як у тебе?");

// Відновлення попереднього стану чату

chat.restore(caretaker.getMemento());

// Вивід стану чату після відновлення

System.out.println("Після відновлення: " + chat.getChatState());

}

}