**LAB ASSIGNMENT 15**

**MEMENTO DESIGN PATTERN**

Memento is a behavioral design pattern that lets you save and restore the previous state of an object without revealing the details of its implementation.

CODE:

* Memento.java

public class Memento {

    private *String* order;

    public Memento(*String* *order*) {

        this.order = order;

    }

    public *String* getOrder() {

        return order;

    }

}

* Originator.java

public class Originator {

    private *String* order;

    public *void* setOrder(*String* *order*) {

        this.order = order;

    }

    public *String* getOrder() {

        return order;

    }

    public *Memento* saveStateToMemento() {

        return new Memento(order);

    }

    public *void* getStateFromMemento(*Memento* *memento*) {

        order = memento.getOrder();

    }

}

* CareTaker.java

import java.util.ArrayList;

import java.util.List;

public class CareTaker {

    private *List*<*Memento*> mementoList = new *ArrayList*<*Memento*>();

    public *void* add(*Memento* *order*) {

        mementoList.add(order);

    }

    public *Memento* get(*int* *index*) {

        return mementoList.get(index);

    }

}

* Client.java

public class Client {

    public static *void* main(*String*[] *args*) {

*Originator* originator = new Originator();

*CareTaker* careTaker = new CareTaker();

        originator.setOrder("Order #1");

        originator.setOrder("Order #2");

        careTaker.add(originator.saveStateToMemento());

        originator.setOrder("Order #3");

        careTaker.add(originator.saveStateToMemento());

        originator.setOrder("Order #4");

        System.out.println("Current Order: " + originator.getOrder());

        originator.getStateFromMemento(careTaker.get(0));

        System.out.println("First saved Order : " + originator.getOrder());

        originator.getStateFromMemento(careTaker.get(1));

        System.out.println("Second saved Order : " + originator.getOrder());

    }

}

OUTPUT:

