# Лабораторная работа № 3. Образцы поведения.

## 3.1 Разработка приложения с использованием паттерна Chain of Responsibility

Реализовать паттерн Chain of Responsibility, обеспечивающий вывод полей объекта типа Transport в текстовый файл в столбик или в одну строку. Для этого нужно разработать интерфейс Chain of Responsibility и два класса-наследника, каждый из которых осуществляет вывод соответствующим образом. В интерфейсе должен быть описан метод записи, в качестве параметра принимающий Транспортное средство, а также метод установки следующего в цепочке. Первая реализация этого интерфейса в цепочке выводит информацию в одну строку, если количество моделей меньше или равно 3. Вторая реализация в цепочке выводит информацию в столбик, если количество моделей больше 3.

Проверить работу паттерна в методе main().

## 3.2 Разработка приложения с использованием паттерна Command

Реализовать паттерн Command, обеспечивающий вывод полей объекта типа Автомобиль в текстовый файл в столбик или в одну строку. Для этого нужно разработать интерфейс Command и два класса-наследника, каждый из которых осуществляет печать соответствующим образом. В классе Автомобиль описать метод print(), которому в качестве параметра передавать поток, куда должна производиться печать. Метод должен обращаться к экземпляру класса, реализующего интерфейс команды (один из двух классов-наследников). Для задания команды добавить метод setPrintCommand() у класса Автомобиль.

Проверить работу паттерна в методе main().

## 3.3 Разработка приложения с использованием паттерна Iterator

Сделать класс Модель в классе Автомобиль доступным на уровне пакета и статическим. Реализовать в нём метод toString(), возвращающий название и цену модели.

Реализовать метод java.util.Iterator iterator() в классе Автомобиль. Для этого следует описать некий дополнительный внутренний класс с некими соответствующими методами (AutoIterator implements java.util.Iterator), экземпляр которого и будет возвращаться методом iterator().

Проверить работу итератора в методе main().

## 3.4 Разработка приложения с использованием паттерна Memento

Реализовать паттерн Memento, обеспечивающий сохранение текущего состояния объекта типа Автомобиль. Для этого нужно разработать соответствующий публичный статический внутренний класс, который будет сохранять состояние текущего объекта в сериализованном виде в массив байт (использовать класс ByteArrayOutputStream) и затем считывать сохраненное состояние. Соответствующие методы назвать setAuto() и getAuto(). В классе Автомобиль описать методы createMemento() и setMemento(), которые будут обращаться к соответствующим методам класса Memento. Проверить работу паттерна в методе main().

## 3.5 Разработка приложения с использованием паттерна Observer

Реализовать приложение, которое рисует на экране «рожицу». При клике мышкой в области глаза глаз должен закрываться (если был открыт) или открываться (если был закрыт). При клике мышкой в области носа его цвет должен измениться. При клике мышкой в области рта рожица должна улыбаться.

## 3.6 Разработка приложения с использованием паттерна State

Реализовать приложение, которое рисует на экране человечка и три кнопки. При нажатии на кнопку «Семестр», человечек должен изменять своё состояние и засыпать. При нажатии на кнопку «Каникулы», человечек должен изменять своё состояние и радоваться. При нажатии на кнопку «Сессия», человечек должен изменять своё состояние и выражать озабоченность или грусть (на ваш выбор).

## 3.7 Разработка приложения с использованием паттерна Strategy

Реализовать паттерн Strategy, обеспечивающий подсчёт количества вхождений каждого элемента в одномерный массив целых чисел двумя разными способами.

Имя входного файла, в котором записан исходный массив в виде сериализованного объекта, передаётся как параметр командной строки приложения.

## 3.8 Разработка приложения с использованием паттерна Template Method

Реализовать многопоточное приложение, которое будет анимировать прыгающий мяч, постоянно перемещая его по экрану и меняя направление движения, когда он встретит преграду в виде стены. Должны быть предусмотрены две кнопки: «Пуск» и «Закрыть». При щелчке по кнопке «Пуск» мяч выбрасывается из правого нижнего угла и начинает прыгать. При каждом нажатии на кнопку «Пуск» добавляется новый мяч. При щелчке по кнопке «Закрыть» приложение завершает свою работу. С помощью паттерна Template Method обеспечить возможность работы приложения не только с мячом, но и с другими фигурами (квадрат, звезда).

Проверить работу паттерна в методе main().

## 3.9 Разработка приложения с использованием паттерна Visitor

Реализовать паттерн Visitor, обеспечивающий печать полей объекта типа Транспортное средство в консоль в столбик или в одну строку. Для этого нужно описать интерфейс Visitor и его реализацию PrintVisitor с двумя вариантами метода visit(), с входным параметром типа Автомобиль (первый метод, выводит всё в одну строку) и Мотоцикл (второй метод, выводит модели и цены в столбик). В интерфейсе Транспортное средство добавить метод accept() с параметром типа Visitor. Каждый из потомков интерфейса Транспортное средство внутри реализации этого метода будет вызывать соответствующий метод visit(). Проверить работу паттерна в методе main().

**Вопросы:**

1. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Chain of Responsibility.
2. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Command.
3. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Interpreter.
4. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Iterator.
5. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Mediator.

6. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Memento.
7. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Observer.
8. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна State.
9. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Strategy.
10. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Template Method.
11. Группа, описание, назначение, область применения, особенности реализации и структурная схема паттерна Visitor.