# Задачи на использование ООП-паттернов проектирования

1. Представьте себе, что вы делаете симулятор животного мира. есть волки, овцы, зайчики, и тп. у них у всех различное поведение. Первая мысль, которая может прийти в голову: давайте сделаем общего предка Animal, от которого уже будут наследоваться всякие там волки, или овцы. наследники будут переопределять какие-то методы, которые отвечают за поведение. волки будут охотиться, овцы - щипать травку и убегать. и тп.

Все бы ничего, но поведение волков неизменно. при таком подходе волки постоянно ведут себя одинаково (как волки). а овцы - как овцы. хотя их поведения и типичны для этих животных, однако оно статичное.

Но вот приходит менеджер и говорит, что заказчикам этого мало. нужно, чтобы в зависимости от времени года, конкретные виды животных так же изменяли свое поведение. например, к зиме волки начинают сбиваться в стаю. а по весне, у животных должны начаться какие-то брачные игры. получается, что в зависимости от времени года одно и тоже животное начинает вести себя по-разному. это означает, что поведение животных **должно уметь изменяться RunTime**. Обычной виртуальной функции здесь уже недостаточно. плюс, менеджеры приносят все новые и новые задачи: **стратегии** поведения животных становятся все более сложными и замороченными.

Нужно решить задачу "иметь возможность изменять поведение налету" ?

1. Разработать класс Светофор, у которого должен быть метот next(), при вызове которого должна загораться текущая секция светофора. Также обязательное условие: добавление секций возможно без переписывания класса Светофор. Это всё можно сделать классически через паттерн состояния. Этот паттерн подразумевает то, что каждое состояние знает куда ему переходить дальше. То есть, если у нас сейчас зелёный, то программа должна понимать, что дальше - жёлтый.

В более сложных версиях у каждого состояния должен быть абстрактный метод не только goToNextState(), но и lightUp(), чтобы реализация поведения была разная