```
/*пример # 35 : классы-наблюдатели : OperationObserver.java : Square.java :
Perimeter.java */
package chapt05.observer;
public abstract class OperationObserver {
    public abstract float valueChanged(Rectangle observed);
}
package chapt05.observer;
public class Perimeter extends OperationObserver {
      private float perimeter;
      public float valueChanged(Rectangle observed) {
            return perimeter =
      2 * (observed.getWidth() + observed.getHeight());
      }
      public String toString() {
            return "P = " + perimeter;
      }
}
package chapt05.observer;
public class Square extends OperationObserver  {
      private float square;
      public float valueChanged(Rectangle observed) {
            return square =
            observed.getWidth() * observed.getHeight();
      }
      public String toString() {
            return "S = " + square;
      }
}
/*пример # 36 : использование шаблона Observer : Main.java */
package chapt05.observer;
public class Main {
    public static void main(String args[]) {
        Rectangle observed = new Rectangle(5, 3);
        System.out.println(observed.toString());
        observed.addObserver(new Square());
        observed.addObserver(new Perimeter());
        observed.setWidth(10);
        System.out.println(observed.toString());
        observed.setHeight(8);
        System.out.println(observed.toString());
    }
}
```

Антишаблоны проектирования

Big ball of mud. «Большой Ком Грязи» – термин для системы или просто программы, которая не имеет хоть немного различимой архитектуры. Как правило, включает в себя более одного антишаблона. Этим страдают системы, разработанные людьми без подготовки в области архитектуры ПО.

Software Bloat. «Распухание ПО» – пренебрежительный термин, используемый для описания тенденций развития новейших программ в направлении использования бóльших объемов системных ресурсов (место на диске, ОЗУ), чем предшествующие версии. В более общем контексте применяется для описания программ, которые используют больше ресурсов, чем необходимо.

Yo-Yo problem. «Проблема Йо-Йо» возникает, когда необходимо разобраться в программе, иерархия наследования и вложенность вызовов методов которой очень длинна и сложна. Программисту вследствие этого необходимо лавировать между множеством различных классов и методов, чтобы контролировать поведение программы. Термин происходит от названия игрушки йо-йо.

Magic Button. Возникает, когда код обработки формы сконцентрирован в одном месте и, естественно, никак не структурирован.

Magic Number. Наличие в коде многократно повторяющихся одинаковых чисел или чисел, объяснение происхождения которых отсутствует.

Gas Factory. «Газовый Завод» – необязательный сложный дизайн или для простой задачи.

Analiys paralisys. В разработке ПО «Паралич анализа» проявляет себя через чрезвычайно длинные фазы планирования проекта, сбора необходимых для этого артефактов, программного моделирования и дизайна, которые не имеют особого смысла для достижения итоговой цели.

Interface Bloat. «Распухший Интерфейс» – термин, используемый для описания интерфейсов, которые пытаются вместить в себя все возможные операции над данными.

Smoke And Mirrors. Термин «Дым и Зеркала» используется, чтобы описать программу либо функциональность, которая еще не существует, но выставляется за таковую. Часто используется для демонстрации финального проекта и его функционала.

Improbability Factor. «Фактор Неправдоподобия» – ситуация, при которой в системе наблюдается некоторая проблема. Часто программисты знают о проблеме, но им не разрешено ее исправить отчасти из-за того, что шанс всплытия наружу у этой проблемы очень мал. Как правило (следуя закону Мерфи), она всплывает и наносит ущерб.

Creeping featurism. Используется для описания ПО, которое выставляет напоказ вновь разработанные элементы, доводя до высокой степени ущербности по сравнению с ними другие аспекты дизайна, такие как простота, компактность и отсутствие ошибок. Как правило, существует вера в то, что каждая новая маленькая черта информационной системы увеличит ее стоимость.

Accidental complexity. «Случайная сложность» – проблема в программировании, которой легко можно было избежать. Возникает вследствие неправильного понимания проблемы или неэффективного планирования.

Ambiguous viewpoint. Объектно-ориентированные модели анализа и дизайна представляются без внесения ясности в особенности модели. Изначально эти модели обозначаются с точки зрения визуализации структуры программы. Двусмысленные точки зрения не поддерживают фундаментального разделения интерфейсов и деталей представления.

Boat anchor. «Корабельный Якорь» – часть бесполезного компьютерного «железа», единственное применение для которого – отправить на утилизацию. Этот термин появился в то время, когда компьютеры были больших размеров. В

настоящее время термин «Корабельный Якорь» стал означать классы и методы,, которые по различным причинам не имеют какого-либо применения в приложении и в принципе бесполезны. Они только отвлекают внимание от действительно важного кода.

Busy spin. Техника, при которой процесс непрерывно проверяет изменение некоторого состояния, например ожидает ввода с клавиатуры или разблокировки объекта. В результате повышается загрузка процессора, ресурсы которого можно было бы перенаправить на исполнения другого процесса. Альтернативным путем является использование сигналов. Большинство ОС поддерживают погружение потока в состояние «сон» до тех пор, пока ему отправит сигнал другой поток в результате изменения своего состояния.

Caching Failure. «Кэширование Ошибки» – тип программного бага (bug), при котором приложение сохраняет (кэширует) результаты, указывающие на ошибку даже после того, как она исправлена. Программист исправляет ошибку, но флаг ошибки не меняет своего состояния, поэтому приложение все еще не работает.

Задания к главе 5

Вариант А

Выполнить описание логики системы и использовать шаблоны проектирования для определения организации классов разрабатываемой системы. Использовать объекты классов и подклассов для моделирования реальных ситуаций и взаимодействий объектов.

1. Создать суперкласс **Транспортное средство** и подклассы **Автомобиль**, **Велосипед**, **Повозка**. Подсчитать время и стоимость перевозки пассажиров и грузов каждым транспортным средством.
2. Создать суперкласс **Пассажироперевозчик** и подклассы **Самолет**, **Поезд**, **Автомобиль**. Задать правила выбора транспорта в зависимости от расстояния и наличия путей сообщения.
3. Создать суперкласс **Учащийся** и подклассы **Школьник** и **Студент**. Определить способы обучения и возможности его продолжения.
4. Создать суперкласс **Музыкальный инструмент** и классы **Ударный, Струнный**, **Духовой**. Определить правила организации и управления оркестром.
5. Создать суперкласс **Животное** и подклассы **Собака**, **Кошка**, **Тигр**, **Мустанг**, **Дельфин**. С помощью шаблонов задать способы обитания.
6. Создать базовый класс **Садовое дерево** и производные классы **Яблоня**, **Вишня**, **Груша**, **Слива**. Принять решение о пересадке каждого дерева в зависимости от возраста и плодоношения.

Тестовые задания к главе 5

Вопрос 5.1.

Какой шаблон создает объекты путем их копирования?

1) Factory;
2) Prototype;
3) Builder;
4) Singleton.