

Паттерны (часть 2)

- 1) Используя паттерн Chain of Responsibility (цепочка обязанностей) разработать систему обработки заказов. Заказ представляет из себя совокупность полей: номер, код товара, количество товара, код заказчика. При этом номер присваивается заказу только после обработки. Каждый заказ может обработать один из множества обработчиков в зависимости от его содержимого. Например, если количество товара больше определенной величины, то его обрабатывают, как оптовый; если заказчик является иностранным поданным, то его обрабатывают по особым условиям; если заказчик является деловым партнером, то его обрабатывают по ускоренной процедуре и т.д. Для различия уже обработанных заказов можно присваивать им разные типы номеров.
- 2) Используя паттерн Command (команда) разработать тестовый язык (например, типа shell), поддерживающий несколько видов команд с возможностью их отмены (undo) и повторения (redo).
- 3) Реализовать паттерн Iterator (итератор) для множества (см. вариант 4 части 1), используя перегрузку операторов ++, *.
- 4) Используя паттерн Adapter (адаптер), реализовать собственную структуру типа стек неограниченного размера, основанную на типе list из стандартной библиотеки. Обязательные методы: push, pop. Также необходимо реализовать собственный класс исключений для случая извлечения элемента из пустого стека.
- 5) Используя паттерн Interpreter, реализовать перевод чисел из цифрового представления в текстовое. Например, на входное число 2381 должно на выходе иметь представление «две тысячи триста восемьдесят один» (диапазон взять хотя бы до миллиона).
- 6) Используя паттерн Proxy, реализовать «умный» указатель для объектов шаблонного класса. Необходимо реализовать подсчет числа ссылок на реальный объект. При отсутствии ссылок память под объект автоматически освобождается.
- 7) Используя паттерн Strategy, реализовать генерацию массива случайных чисел (равномерное распределение) не менее, чем 3-мя разными алгоритмами. Параметры генерации: границы отрезка и объем выборки.
- 8) Используя паттерн Observer, реализовать модель электронного аукциона. Пул товаров (лотов) фиксирован, число участников определяется на этапе выполнения, при этом оно может изменяться в зависимости от лота.
- 9) Реализовать защищающий Proxy. Рассмотрим упрощенный каталог электронных ресурсов НИУ ВШЭ. Из внутренней сети Вышки студенты и сотрудники могут получить доступ к списку журналов и статей, а также скачивать тексты этих статей. Из внешней сети можно получить лишь список журналов. Таким образом, интерфейс «Электронная библиотека НИУ ВШЭ» состоит из операций: получить список журналов и скачать текст статьи. Есть также интерфейс читателя, в котором есть методы, возвращающие уровень доступа. Интерфейс читателя реализуется по-разному для читателя из внутренней сети и читателя из внешней сети.

10) Реализовать паттерн "Стратегия" для сущности "Дверь". Определить 2 различных аспекта поведения. Например: "Имеет замок", "Открывается". Создать интерфейсы для этих аспектов. Интерфейс должен содержать как минимум 1 метод. Создать по 2 конкретных класса, реализующих каждый интерфейс поведения. Каждый класс должен иметь конкретные реализации метода(ов), унаследованного(ых) от интерфейса и выводить результат поведения.