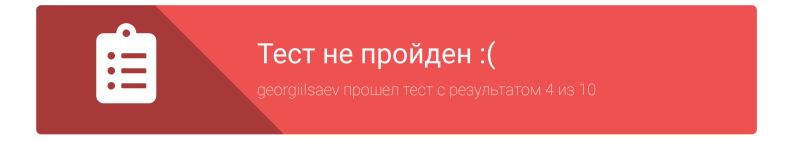
## ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ 🔗



### РЕЗУЛЬТАТЫ

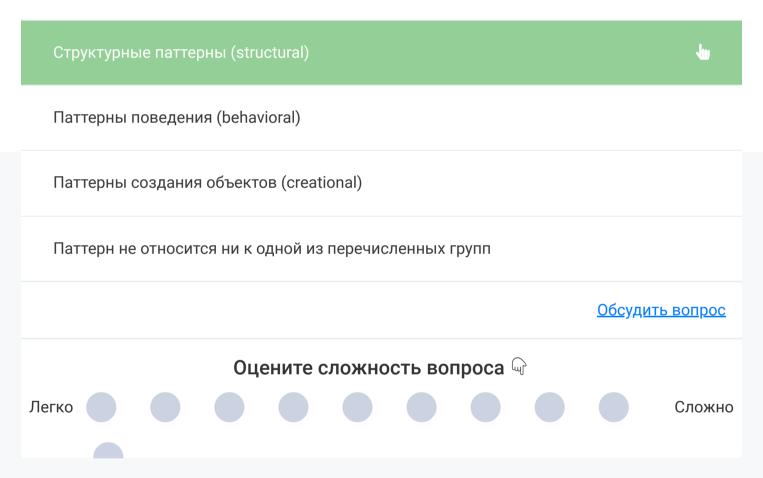
16
Заработанные очки



**40%**Правильных ответов

### **ОТВЕТЫ**

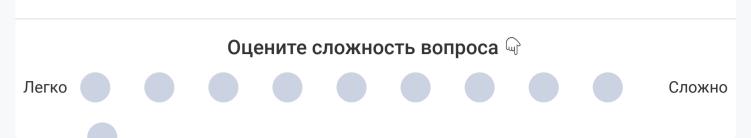
#1 К какой группе относится паттерн проектирования "Decorator" (в соответствии с GoF)?



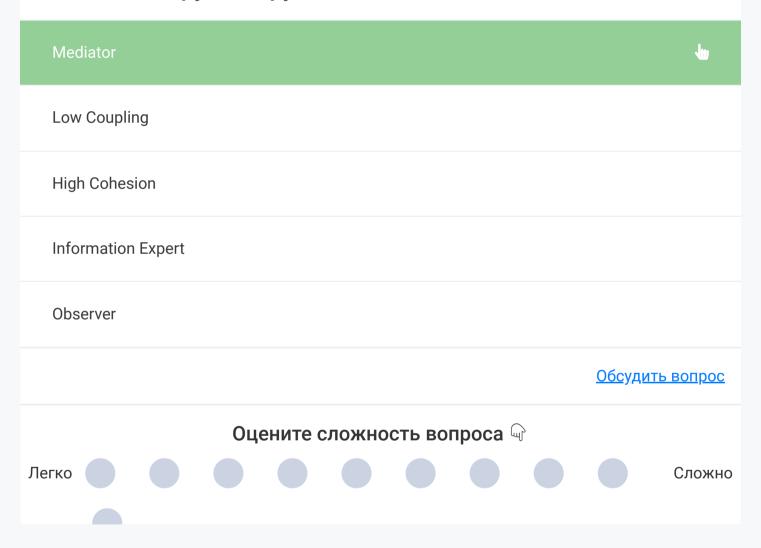


**Объяснение**: Паттерн Фасад — это структурный паттерн проектирования, предоставляющий простой интерфейс к сложной системе классов, библиотеке или фреймворку, а Адаптер позволяет объектам с несовместимыми интерфейсами работать вместе.

Фасад задаёт новый интерфейс, тогда как Адаптер повторно использует старый. Адаптер оборачивает только один класс, а Фасад оборачивает целую подсистему. Более того, Адаптер позволяет двум существующим интерфейсам работать сообща, вместо того, чтобы задать полностью новый.



#3 Какой шаблон проектирования можно определить следующей фразой: ".. - поведенческий шаблон проектирования, обеспечивающий взаимодействие множества объектов, формируя при этом слабую связанность и избавляя объекты от необходимости явно ссылаться друг на друга "?



# #4 Какие из перечисленных утверждений верны по отношению к паттерну проектирования Адаптер (Adapter):

Паттерн преобразует интерфейс одного класса в интерфейс другого, который ожидают клиенты.



Паттерн определяет представителя другого объекта, не изменяя его интерфейс.

Структура паттерна Адаптер (Adapter) аналогична структуре паттерна Мост (Bridge).

**Объяснение**: 1. Структура паттерна мост анологична структуре адаптера, но у моста иное назначение. Он отделяет интерфейс от реализации, чтобы то и другое можно было изменять независимо. Адаптер же призван изменить интерфейс существующего объекта. GoF

2. Замеситель (Proxy) определяет представителя или суррогат другого объекта, но не изменяет его интерфейс. GoF

#### Оцените сложность вопроса 🖓

Легко



#5 Поведенческий паттерн проектирования, который позволяет передавать запросы последовательно по цепочке обработчиков. Каждый последующий обработчик решает, может ли он обработать запрос сам и стоит ли передавать запрос дальше по цепи. Какой это паттерн?

компоновщик (composite)

цепочка обязанностей (chain of responsibility)



наблюдатель (observer)

Обсудить вопрос

**Объяснение**: Цепочка обязанностей (англ. CoR, Chain of Command, Chain of Responsibility) - паттерн который представляет из себя поведенческий паттерн проектирования, позволяющий передавать запросы последовательно по цепочке обработчиков. Каждый последующий обработчик решает, может ли он обработать запрос сам и стоит ли передавать запрос дальше по цепи.

#### Оцените сложность вопроса 🖓

Легко











#6 Какие основные задачи решает применение паттернов проектирования?

Производится унификация терминологии, названий модулей и элементов проекта.

Показываются отношения и взаимолействия между классами или объектами.

Облегчается дискуссия об абстрактных структурах данных между разработчиками. 👆



Описывается решение целого класса абстрактных проблем.



Объяснение: Паттерны проектирования — это многократно применяемая архитектурная конструкция, предоставляющая решение общей проблемы проектирования в рамках конкретного контекста и описывающая значимость этого решения. Паттерн не является законченным образцом проекта, который может быть прямо преобразован в код (что отличает его от близкого понятия идиомы), скорее это описание или образец для того, как решить задачу, таким образом, чтобы это можно было использовать в различных ситуациях. Объектно-ориентированные паттерны зачастую показывают отношения и взаимодействия между классами или объектами, без определения того, какие конечные классы или объекты приложения будут использоваться.

Главная польза каждого отдельного паттерна состоит в том, что он описывает решение целого класса абстрактных проблем. Также тот факт, что каждый паттерн имеет свое имя, облегчает дискуссию об абстрактных структурах данных (ADT)

между разработчиками, так как они могут ссылаться на известные шаблоны. Таким образом, за счёт паттернов производится унификация терминологии, названий модулей и элементов проекта.

# Оцените сложность вопроса ♀ Легко Сложно

## #7 Какая разница между паттернами Facade и Adapter?

Аdapter оборачивает один класс, тогда как Facade - несколько.

Аdapter изменяет интерфейс класса так, чтобы он удовлетворял требованиям пользователя, а Facade предоставляет унифицированный интерфейс к некоторому множеству классов.

Паттерн Adapter убирает зависимость между клиентской реализацией и интерфейсом системы, а Facade вынуждает пользователя реализовывать определенный интерфейс.

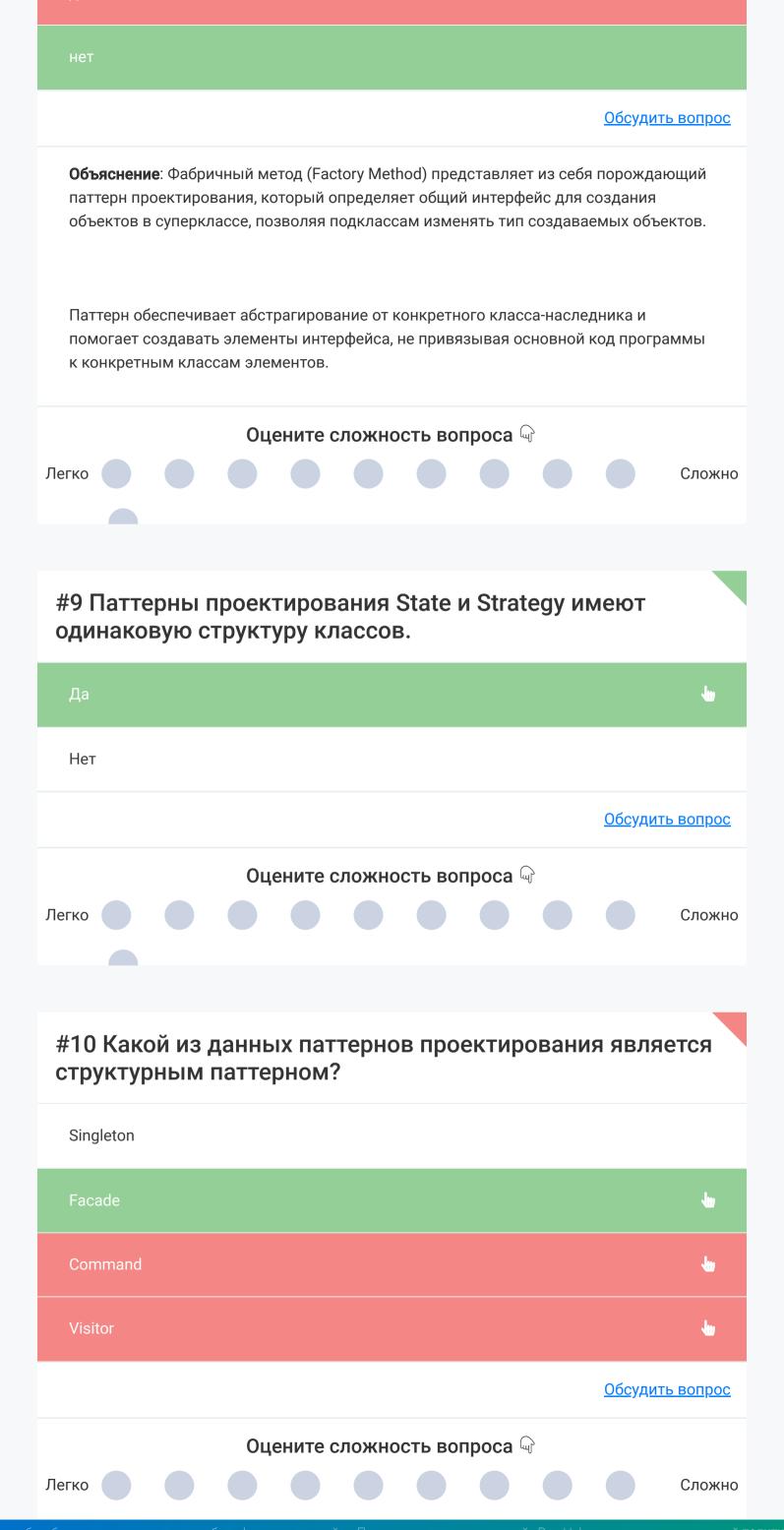
Оба паттерна это разновидности одного и того же общего случая.

Ни один из перечисленных ответов не является верным.

Обсудить вопрос

Оцените сложность вопроса Ф

#8 Паттерн проектирования Factory Method обеспечивает тесное связывание между созданным объектом и использующим его кодом.





support@mg.proghub.ru