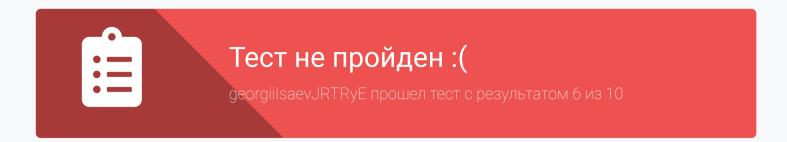
ПАТТЕРНЫ ПРОЕКТИРОВАНИЯ 🔗



РЕЗУЛЬТАТЫ

24
Заработанные очки



60%

ОТВЕТЫ

#1 Целью какого паттерна является расширение функциональности класса или же ее изменение без использования механизма наследования?

Proxy

Decorator

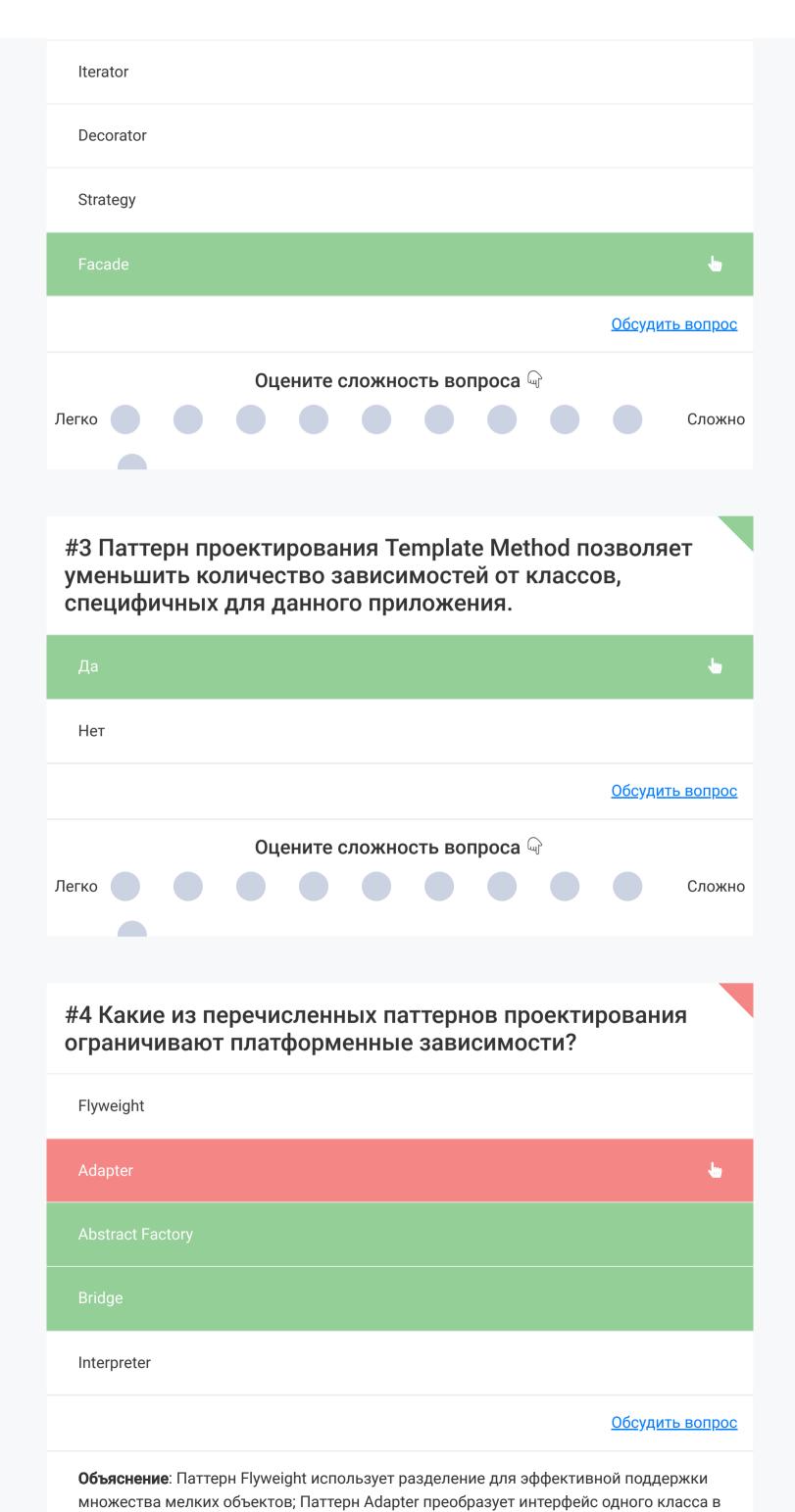
Flyweight

Composite

Объяснение: Структурный паттерн Decorator используется в случаях, когда необходимо без применения механизма наследования расширить функциональность класса или же изменить ее. Другими словами появляется альтернатива наследованию, причем классы не закреплены жестко в иерархии.

Оцените сложность вопроса
Оцените сложность вопроса
Оцените сложность вопроса
Сложно

#2 Назначением какого паттерна проектирования является предоставление удобного интерфейса к громоздкому и сложному API?



интерфейс другого, который ожидают клиенты; Паттерн Interpreter для заданного языка определяет представление его грамматики, а также интерпретатор предложений этого языка; Паттерны Abstract Factory и Bridge отделяют абстракцию от ее реализации так, чтобы то и другое можно было изменять независимо. Это обеспечивает кроссплатформенность.

Оцените сложность вопроса Ф Легко Сложно

#5 Какие из перечисленных утверждений верны по отношению к паттерну проектирования Composite:

Composite паттерн содержит в себе несколько моделей поведения объекта и предоставляет возможность переключаться между ними.

Паттерн Composite позволяет работать с композитными и простыми объектами одинаковым образом.

Паттерн Composite организовывает объекты в древовидные структуры.

4

Composite паттерн упрощает работу с группой объектов не связанных типов.

Обсудить вопрос

Оцените сложность вопроса 🖓

Легко















Сложно

#6 Какая разница между паттернами Facade и Adapter?

Adapter оборачивает один класс, тогда как Facade - несколько.

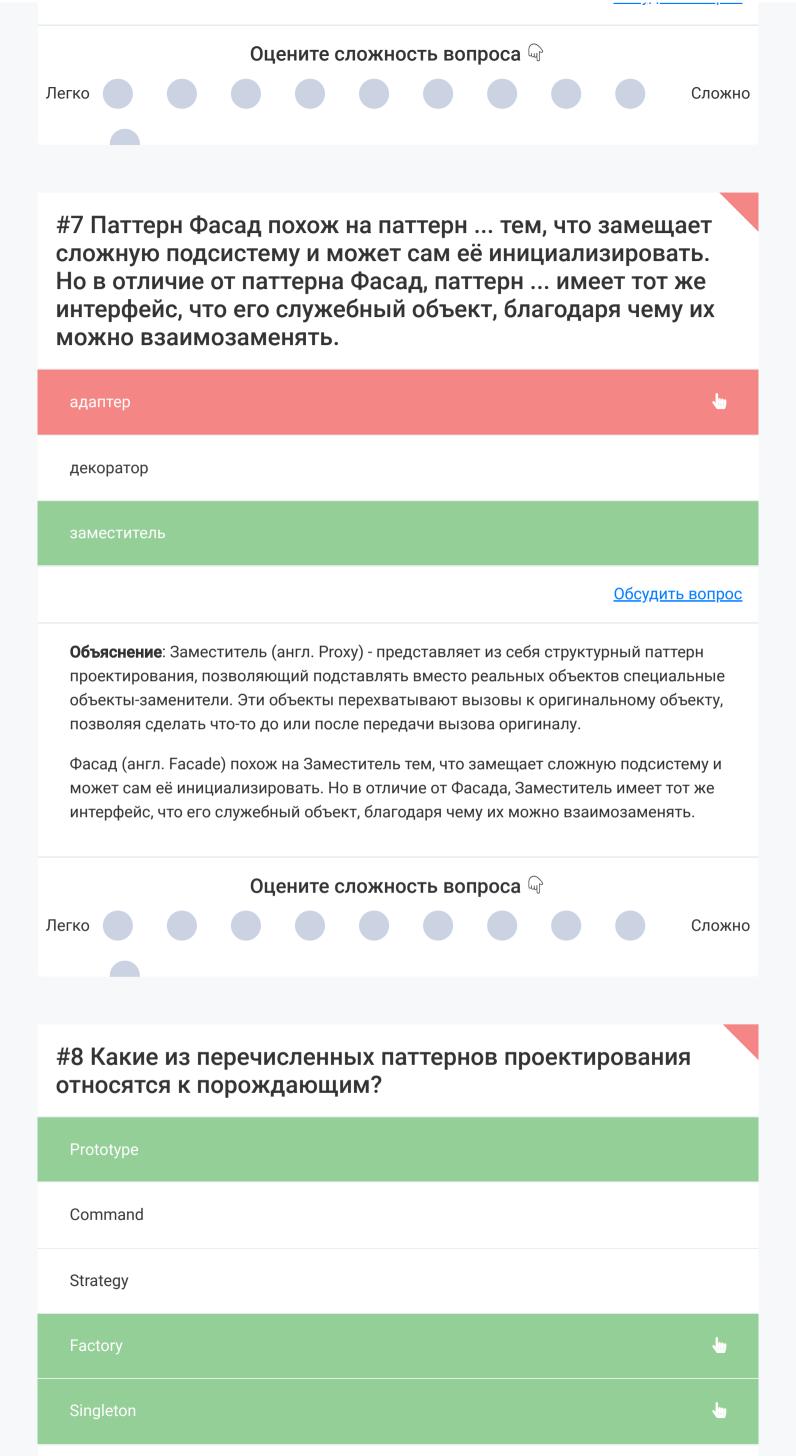
Adapter изменяет интерфейс класса так, чтобы он удовлетворял требованиям пользователя, а Facade предоставляет унифицированный интерфейс к некоторому



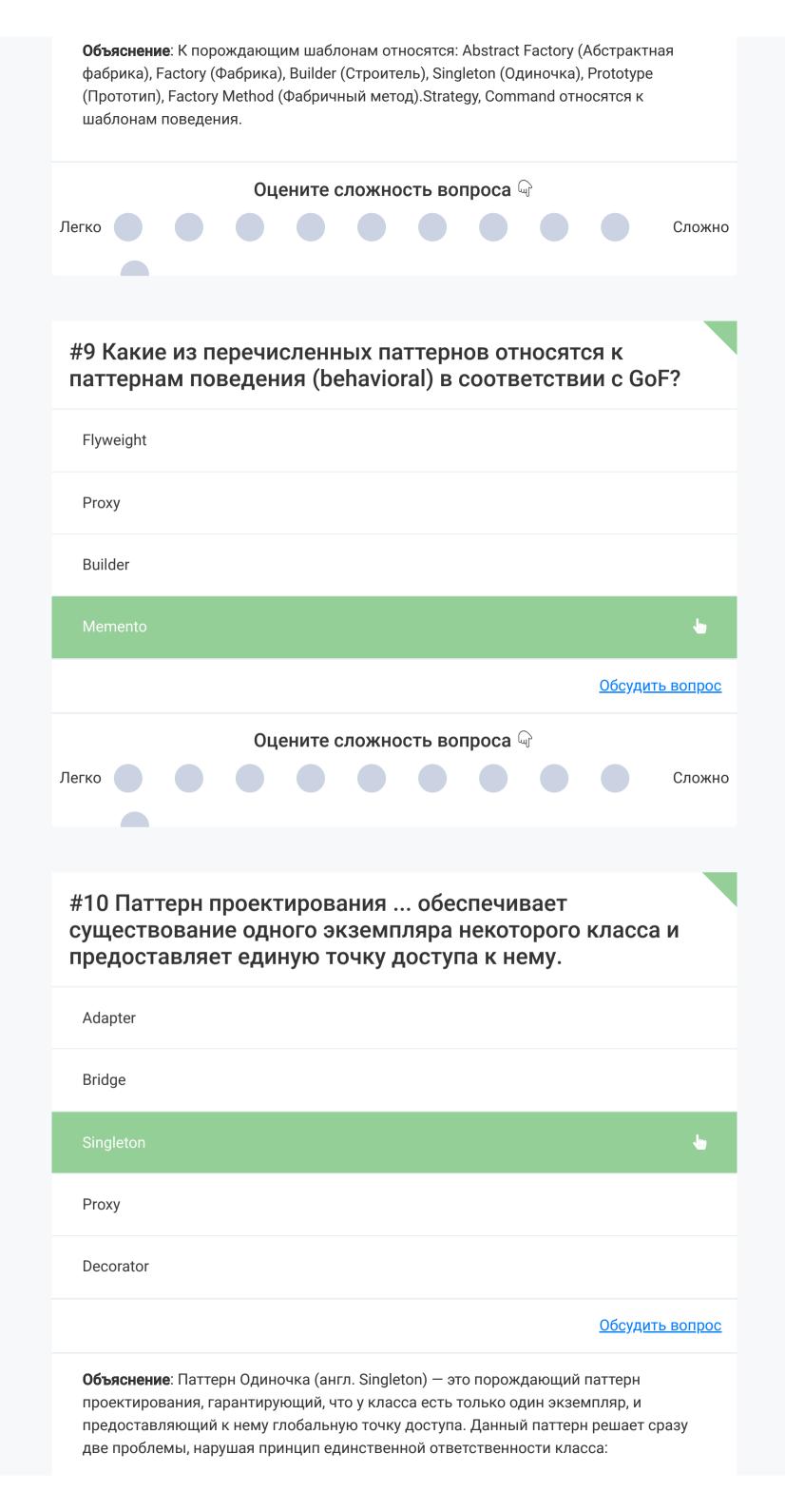
Паттерн Adapter убирает зависимость между клиентской реализацией и интерфейсом системы, а Facade вынуждает пользователя реализовывать определенный интерфейс.

Оба паттерна это разновидности одного и того же общего случая.

Ни один из перечисленных ответов не является верным.



Обсудить вопрос



Гарантирует наличие единственного экземпляра класса. Чаще всего это полезно для доступа к какому-то общему ресурсу, например, базе данных.
 Предоставляет глобальную точку доступа. Это не просто глобальная переменная, через которую можно достучаться к определённому объекту. Глобальные переменные не защищены от записи, поэтому любой код может подменять их значения без вашего ведома.
 Оцените сложность вопроса Прегко
 Одените сложность вопроса Прегко

Осложно

support@mg.proghub.ru