Вариант 1. Абстрактная фабрика

Вы создаете библиотеку для игры-стратегии. Три различные армии – русская, английская, немецкая. Каждая армия имеет подразделения – пехота, кавалерия, артиллерия. Игрок может выбрать армию, за которую будет играть, а также противника для игры. Каждое подразделение характеризуется параметрами – защита, атака, скорость (задать произвольно). Каждое подразделение также имеет ранг – от 0 до 4. Подразделение можно повысить в ранге и получить текущий ранг подразделения. Необходимо реализовать процесс добавления подразделения в армию и удаление подразделения из армии.

Вариант 2. Строитель

Вы создаете построитель стартовой страницы сайта. Создается отдельно блок head, блок body, а также внутри блока body: логотип-блок, контент (просто текст), «футер» («подвал»). Несколько компаний будет пользоваться стартовой страницей: SuperCompany, JustCompany, Company. У каждой есть год основания, что они бы хотели отобразить в футере (при этом JustCompany не хочет отображать). Логотип для SuperCompany отображается жирным тестом, а JustCompany - курсивом

Вариант 3. Одиночка, Dependency Injection

Нужно создать репозиторий, который будет получать данные из текстового файл, а также есть тестовый репозиторий, который получает данные прямо из кода (из переменной). Также у вас есть логгер, который заносит данные в текстовый файл и тестовый логгер, который просто отображает данные на консоли. Для показа работы данных классов необходимо использовать «внедрение зависимости» для класса

Вариант 4. Адаптер

Написать реализацию климат-контроля (который просто возвращает значения температуры и т.д., которые мы передадим в конструкторе. При этом температура в фаренгейтах, давление в паскалях, также имеется возможность изменять данные температуры и давления.

Представим, что прошел год. Нам понадобился климат-контроль, который работает с температурой по цельсию и давлением в мегапаскалях. Написать адаптер к уже имеющейся реализации климат-контроля

Вариант 5. Мост, Dependency Injection

Вы пишете логгер, который может заносить логи, которые могут быть следующего вида: информация, предупреждение, ошибка. При этом существует два вида логгеров: один заносит все в файл, другой на консоль. Кроме того существует несколько типов логгеров: первый перед каждым логом ставит время и двоеточие, второй – перед каждым логом ставит стрелочку в виде двух символов: ->, а третий ничего не ставит. Для показа работы данных классов необходимо использовать «внедрение зависимости» класса

Вариант 6. Компоновщик

Вы пишете библиотеку для работы с файловой системой. Необходимо реализовать следующее: возможность создания произвольной структуры папок и файлов. Получение информации о файле или папке. Добавление и удаление папок и файлов в папке. Для работы вам не нужно использовать реальную файловую систему, вы просто создаете абстрактную папку, в которую добавляете что-то (в коде, на реальной файловой системе это не должно отображаться)

Вариант 7. Декоратор, Dependency Injection

У вас есть класс, который заносит какой-то задачи в список, и при необходимости может отображать его.

Через год, вам понадобилось добавить функциональность для него: некоторые списки вы хотите отображать так, чтобы каждая задача помещалась в кавычки, другие списки так, чтобы между каждой задачей помещались символы ------ на отдельной строке, для третьих, чтобы в конце списка отображалась надпись: «Всего: 5 задач». Кроме того, иногда вам требуются любые комбинированные списки (например где будет отображаться и кавычки и надпись после списка )

Вариант 8. Фасад, Dependency Injection

Необходимо создать систему следующих классов: один, сохраняет и возвращает температуру, другой делает то же самое с давлением, третий – параметры ветра (направлние и силу), четвертый – влажность воздуха. Необходимо создать класс, который возвращает данные в виде текста, например: «Температура сегодня +3 градуса по цельсию, давление 755 мм.рт.столба ...». Также необходим еще один класс, который не только возвращает такой текст, но и выводит его на экран

Вариант 9. Приспособленец

Необходимо создать псевдографический построитель (т.е. если у вас хранятся координаты центра круга, его радиус, толщина линии, цвет линии, заливка, то просто вывести на консоль: «Рисую круг с центром в точке (1, 0) толщиной 5 пикселей синего цвета с заливкой красного цвета»)

Вы должны хранить группы фигур (при этом можно добавлять группу и удалять). В каждой группе хранятся фигуры, которой может быть – прямоугольник, круг, квадрат. Параметры внутри группы должны быть одинаковыми – толщина линии, цвет, заливка.

Вариант 10. Прокси, Dependency Injection

Создать класс, который содержит данные о пользователе (имя, фамилия, возраст). Напрямую обращаться к этому классу нельзя. Спроектировать несколько прокси (на вход прокси получает логин/пароль). Первый – записывает в файл все попытки получить данные о пользователе, но ничего не возвращает. Второй – возвращает данные, только если логин/пароль верны (корректный логин/пароль можно выбрать произвольно).

Вариант 11. Цепочка обязанностей

Вы создаете простой таск-менеджер для документов. Имеется массив задач, каждая из которых состоит из двух составляющих – названия файла и пометки («удалить», «переместить в важные», «переместить в черновики», «дописать в конце файла подпись»). При этом не обязательно работать с реальной файловой системой. Можно показать работу на примере хранения документов в памяти.

++Вариант 12. Состояние, Dependency Injection

Вы создаете текстовый редактор, который может находится в трех состояниях – только для чтения, режим редактирования и режим только для чтения всех, кроме администратора (из этого режима можно перейти в другой только введя пароль администратора, не нужно привязываться к реальному администратору, можно выбрать произвольный пароль). Через Dependency Injection можно указать начальное состояние.

Вариант 13. Стратегия, Dependency Injection

Вы создаете конвертер документа к какому-либо виду. Существует несколько вариантов преобразования – все символы в верхнем регистре, все символы в нижнем регистре, после точки первый символ в верхнем, остальные в нижнем. Через Dependency Injection выбираем текущий конвертер. Реальные документы можно не обрабатывать, можно просто обрабатывать текст в памяти.

Вариант 14. Шаблонный метод, Dependency Injection

Вы создаете построитель стартовой страницы сайта. Несколько компаний будет пользоваться стартовой страницей: SuperCompany, JustCompany, Company. У каждой есть год основания, что они бы хотели отобразить в футере (при этом JustCompany не хочет отображать). Логотип для SuperCompany отображается жирным тестом, а JustCompany курсивом. Через Dependency Injection выбираем текущую компанию.