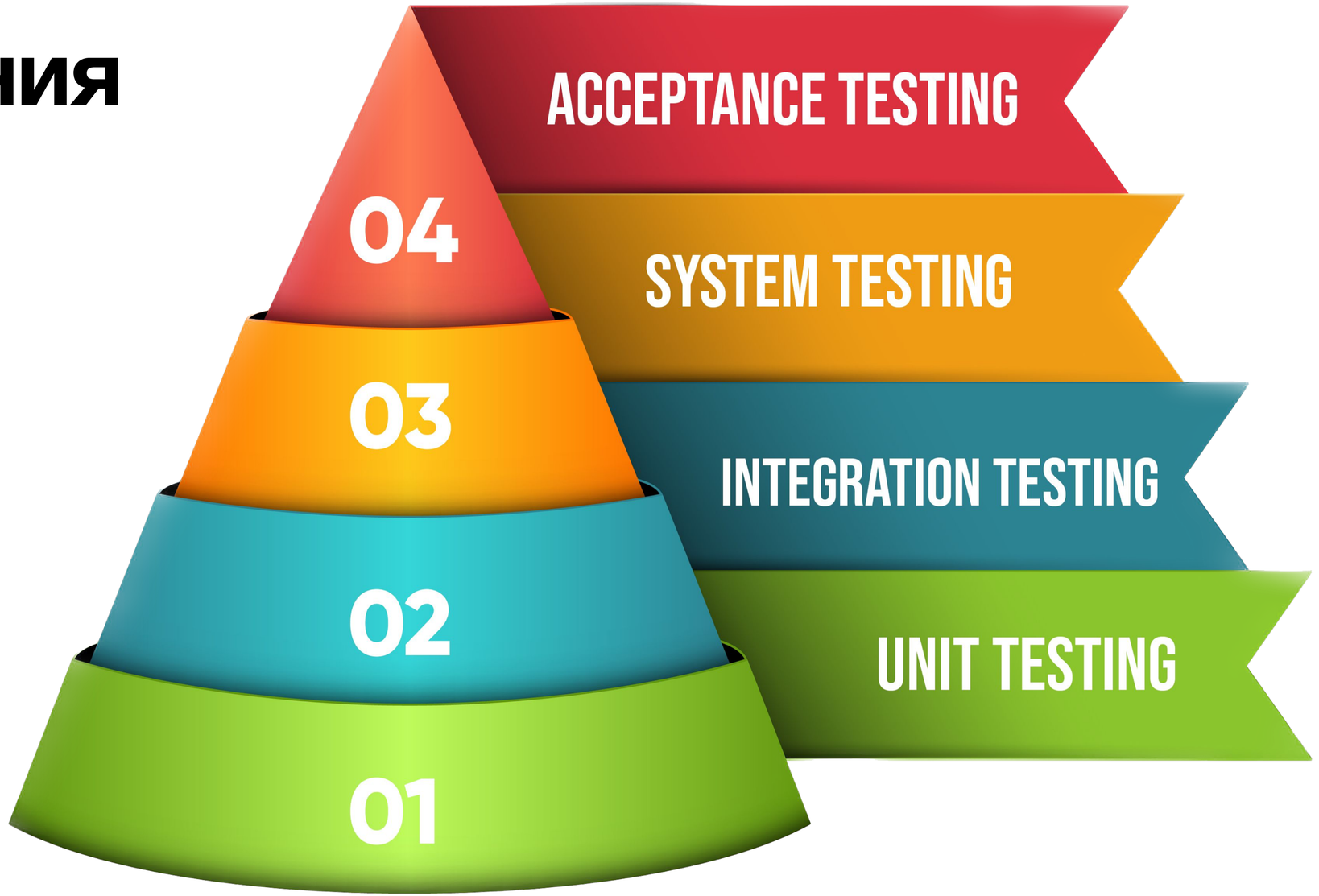
Тестирование

Unit-тестирование, моки

Неделя 4

Уровни тестирования



Unit-тестирование

Цель:

- Найти дефекты в компонентах (классы/модули/функции)
- Сформировать уверенность *(на каком-то уровне)*, что компонент работает
- Заложить базу для следующих уровней тестирования

На вход:

- Код
- Модель данных, спецификация, описание компонента…

Unit-тестирование

Объекты тестирования:

- Классы
- Компоненты, модули

Типичные ошибки/дефекты:

- Проблемы с потоком данных и потоком управления
- Неверная логика
- Не вписываемся в требования

Интеграционное тестирование

Цель:

- Найти сбои в взаимодействии разных интерфейсов
- Сформировать уверенность в качестве интерфейсов *(на каком-то уровне)*
- Избежать ошибок на уровнях выше
- Найти дефекты нижнего уровня

На вход:

- Спецификация протоколов, описание интерфейсов
- Описание архитектуры, диаграмма последовательности, варианты использования

Интеграционное тестирование

Объекты тестирования:

- Компоненты, модули
- Микросервисы
- API

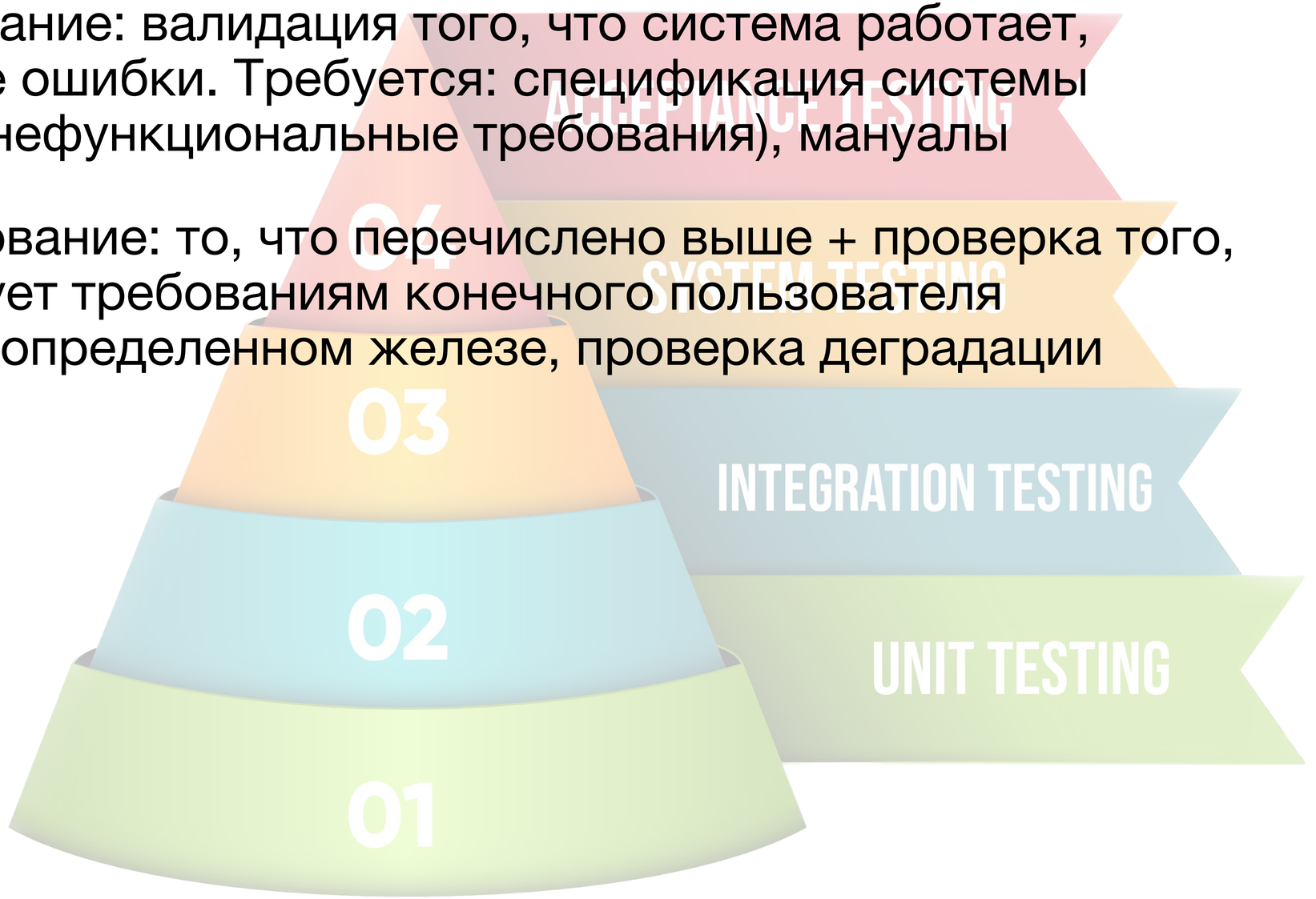
Типичные ошибки/дефекты:

- Пересылаются неправильные сообщения/данные
- Несоответствие между интерфейсом отправителя и принимающего

Тестирование системы, приемочное тестирование

- Системное тестирование: валидация того, что система работает, находим глобальные ошибки. Требуется: спецификация системы (функциональные и нефункциональные требования), мануалы

- Приемочное тестирование: то, что перечислено выше + проверка того, система соответствует требованиям конечного пользователя (тестирование на на определенном железе, проверка деградации функциональности)



Фреймворки для unit тестирования

- xUnit (рассмотрим сейчас)
- MSTest
- nUnit

Для моков:

- NSubstitute
- Moq

+ AutoFixture, FluentAssertions

Настройка проекта xUnit

Для создания проекта с поддержкой xUnit необходимо выполнить следующие шаги:

1. Открыть терминал и перейти в директорию с решением.
2. Выполнить команду `dotnet new xunit -n UniversalCarShop.Tests`.
3. Выполнить команду `dotnet sln add UniversalCarShop.Tests/UniversalCarShop.Tests.csproj`.

После выполнения этих шагов в директории решения появится новый проект с поддержкой xUnit, в котором уже будет добавлен файл с тестом UnitTest1.cs. Данный файл можно удалить, так как он не будет использоваться.

Чтобы мы могли тестировать классы из основного проекта, необходимо добавить ссылку на него в тестовый проект.

Для этого в терминале необходимо:

1. Перейти в директорию с тестовым проектом.
2. Выполнить команду `dotnet add reference ../UniversalCarShop/UniversalCarShop.csproj`.

Здесь вместо `../UniversalCarShop/UniversalCarShop.csproj` необходимо указать путь к основному проекту.

Задание для решения

1. Проверьте, что метод `IsCompatible` класса `PedalEngine` не учитывает силу рук покупателя.
2. Проверьте, что свойство `Number` класса `Car` содержит значение, которое было передано при создании объекта.
3. Проверьте, что метод `GetCustomers` класса `CustomersStorage` возвращает добавленных покупателей.
4. Проверьте, что метод `SellCars` класса `HseCarService` продает автомобили добавленным покупателям.
5. Проверьте, что метод `SellCars` класса `HseCarService` не продает один и тот же автомобиль двум покупателям.