***Уровни тестирования***

**Компонентное тестирование Модульное тестирование UNIT-Testing** - процесс в [программировании](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5), позволяющий проверить на корректность отдельные модули [исходного кода](https://ru.wikipedia.org/wiki/%D0%98%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4) программы.

*Цель* модульного тестирования — изолировать отдельные части программы и показать, что по отдельности эти части работоспособны. Оптимальный вариант подхода – автоматизированное тестирование. Этот тип тестирования обычно выполняется [программистами](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%81%D1%82).

*Базис тестирования*:

· Требования к компонентам

· Детальный дизайн

· Код

*Обьекты тестирования*:

· Компоненты

· Программы

· Программы конвертации и миграции данных

· Модули БД

**Интеграционное тестирование -** одна из фаз тестированияпрограммного обеспечения, при которой отдельные программные модули объединяются и тестируются в группе. Обычно интеграционное тестированиепроводится после модульного тестирования и предшествует системномутестированию.

*Уровни интеграционного тестирования***:**

* **Компонентный интеграционный уровень** (*Component Integration testing*)

Проверяется взаимодействие между компонентами системы после проведения компонентного тестирования.

* **Системный интеграционный уровень** (*System Integration Testing*)

Проверяется взаимодействие между разными системами после проведения системного тестирования.

*Подходы к интеграционному тестированию***:**

* **Снизу вверх** (*Bottom Up Integration*)

Все низкоуровневые модули, процедуры или функции собираются воедино и затем тестируются. После чего собирается следующий уровень модулей для проведения интеграционного тестирования. Данный подход считается полезным, если все или практически все модули, разрабатываемого уровня, готовы. Также данный подход помогает определить по результатам тестирования уровень готовности приложения (см. также [Integration testing - Bottom Up](http://en.wikipedia.org/wiki/Integration_testing#Bottom_Up))

* **Сверху вниз** (*Top Down Integration*)

Вначале тестируются все высокоуровневые модули, и постепенно один за другим добавляются низкоуровневые. Все модули более низкого уровня симулируются заглушками с аналогичной функциональностью, затем по мере готовности они заменяются реальными активными компонентами. Таким образом мы проводим тестирование сверху вниз. (см. также [**Top Down Integration**](http://en.wikipedia.org/wiki/Integration_testing#Top-down_and_Bottom-up))

* **Большой взрыв** (*"Big Bang" Integration*)

Все или практически все разработанные модули собираются вместе в виде законченной системы или ее основной части, и затем проводится интеграционное тестирование. Такой подход очень хорош для сохранения времени. Однако если тест кейсы и их результаты записаны не верно, то сам процесс интеграции сильно осложнится, что станет преградой для команды тестирования при достижении основной цели интеграционного тестирования (см. также [**Integration testing - Big Bang**](http://en.wikipedia.org/wiki/Integration_testing#Big_Bang))

*Базис тестирования*:

· Проект системы

· Архитектура

· Бизнес-процессы

· Сценарии использования

*Обьекты тестирования*:

· БД подсистемы

· Инфраструктура

· Интерфейсы

· Конфигурация системы

· Конфигурационные данные

**Системное тестирование** - это [тестирование программного обеспечения](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F) (ПО), выполняемое на полной, интегрированной системе, с целью проверки соответствия системы исходным требованиям. Системное тестирование относится к методам [тестирования чёрного ящика](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%87%D1%91%D1%80%D0%BD%D0%BE%D0%B3%D0%BE_%D1%8F%D1%89%D0%B8%D0%BA%D0%B0), и, тем самым, не требует знаний о внутреннем устройстве системы.

*Типы системных тестов*:

·  [Функциональное тестирование](https://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D0%B5_%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

·  [Тестирование пользовательского интерфейса](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D1%81%D0%BA%D0%BE%D0%B3%D0%BE_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D0%B0)

·  [Юзабилити-тестирование](https://ru.wikipedia.org/wiki/%D0%AE%D0%B7%D0%B0%D0%B1%D0%B8%D0%BB%D0%B8%D1%82%D0%B8-%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

·  [Тестирование совместимости](https://ru.wikipedia.org/w/index.php?title=%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%81%D0%BE%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%B8%D0%BC%D0%BE%D1%81%D1%82%D0%B8&action=edit&redlink=1)

·  [Тестирование на основе модели](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BD%D0%B0_%D0%BE%D1%81%D0%BD%D0%BE%D0%B2%D0%B5_%D0%BC%D0%BE%D0%B4%D0%B5%D0%BB%D0%B8)

·  [Тестирование безопасности](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D0%B8)

·  [Тестирование производительности](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%BE%D0%B4%D0%B8%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D0%B8)

·  [Регрессионное тестирование](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B3%D1%80%D0%B5%D1%81%D1%81%D0%B8%D0%BE%D0%BD%D0%BD%D0%BE%D0%B5_%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

·  [Автоматическое тестирование](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D1%82%D0%BE%D0%BC%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%BE%D0%B5_%D1%82%D0%B5%D1%81%D1%82%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

*Два подхода к системному тестированию*

* на базе требований (*requirements based*)

Для каждого требования пишутся [тестовые случаи (test cases)](http://www.protesting.ru/testing/testcase.html), проверяющие выполнение данного требования.

* на базе случаев использования (*use case based*)

На основе представления о способах использования продукта создаются случаи использования системы (Use Cases). По конкретному случаю использования можно определить один или более сценариев. На проверку каждого сценария пишутся [тест кейсы (test cases)](http://www.protesting.ru/testing/testcase.html), которые должны быть протестированы.

**Приемочное тетстирование** - вид [тестирования ПО](http://qatestlab.com/ru/company/why-software-testing/), который проводится с целью проверить, разработан ли продукт в соответствии со стандартами, заданными критериями и отвечает ли всем требованиям заказчика.

## *Цели приемочного тестирования***:**

· Определить, соответствует ли система приемочным критериям

· Вынести решение: отправлять на доработку или принять приложение

## *Преимущества приемочного тестирования для клиентов:*

· Приемочное тестирование помогает отыскать баги, связанные с удобством простотой приложения для пользователей

· Данный вид тестирования является финальным этапом перед запуском приложения

· Приемочное тестирование осуществляется с помощью реального восприятия приложения конечными пользователем.

**Классификация тестирования**

**I**

1) *Black box testing* - тестирование, когда мы не знаем ничего о системе, внешней части ПО, например интерфейс

2) *White box* - более углубленные знания, возможность тестирования внутренней части(кода, элементов)

3) *Grey box* - тестирование включающее знания как белого так черного, понемногу.

**II**

1) *Альфа тестирование*- когда продукт тестируется на территории разработчика

2) *Бета тестирование*- когда продукт тестируется заказчиком на его территории в его офисе

**III**

1) *Динамическое тестирование* - тестирование требований и кода (нету готового продукта)

2) *Статическое* - при получении данных с кода можем тестировать

**IV**

1)*Позитивное тестирование* - тестирование системы на введение корректных данных, валидных значений, для проверки выполняет ли ПО свою основную функциональность, для чего она создавалась

2) *Негативное* - проверка ПО на ввод неверных и не корректных данных, анализ результата(баги)

**V**

1)*ручное тестирование*

2)*автоматическое*