1. **Процесс тестирования** — это последовательность шагов, которые выполняются для проверки качества программного обеспечения. Он включает в себя планирование, создание тестовых случаев, выполнение тестов, анализ результатов и отчетность. Цель процесса тестирования — обнаружить дефекты и убедиться, что приложение работает корректно.
2. **TestNG** и **JUnit** — это фреймворки для автоматизированного тестирования в Java:
   * **TestNG** (Test Next Generation) — это более гибкий и мощный фреймворк, который позволяет создавать сложные тестовые сценарии, управлять зависимостями между тестами и обеспечивать параллельное выполнение тестов.
   * **JUnit** — более простой фреймворк, который используется для модульного тестирования. Он предоставляет аннотации для создания тестовых методов и проверки ожидаемых результатов.
3. **Юнит-тестирование** — это проверка отдельных компонентов (классов, методов) программы на корректность их работы. Основные признаки юнит-тестирования:
   * Тестирование изолированных частей кода (без зависимостей).
   * Автоматизированное выполнение тестов.
   * Проверка ожидаемых результатов.
4. **Тест-кейс** — это документ, описывающий шаги для выполнения конкретного теста. Он включает в себя предусловия, шаги, ожидаемые результаты и постусловия.
5. Аннотация **@Test** используется в TestNG и JUnit для обозначения методов, которые должны быть выполнены как тестовые. Эти методы будут запускаться автоматически во время тестирования.
6. **Test suite** — это группа тестов, которые могут быть выполнены вместе. Для создания test suite в TestNG, вы можете использовать XML-файл, в котором перечислены тестовые классы.
7. **Assert class** обеспечивает проверку условий во время выполнения тестов. Он используется для проверки ожидаемых результатов и генерации ошибок, если результаты не соответствуют ожиданиям.
8. Аннотации **@BeforeSuite**, **@AfterSuite**, **@BeforeTest**, **@AfterTest**, **@BeforeClass**, **@AfterClass**, **@BeforeMethod** и **@AfterMethod** используются в TestNG для выполнения дополнительных действий перед и после выполнения тестов. Например, **@BeforeSuite** выполняется перед всеми тестами в suite, а **@BeforeMethod** перед каждым тестовым методом.
9. Параметризация в тестировании позволяет передавать разные наборы данных в один и тот же тестовый метод. Это полезно, например, для проверки разных входных значений или сценариев.
10. **TDD** (Test-Driven Development) — это методология разработки, при которой тесты создаются перед написанием кода. **DDT** (Data-Driven Testing) — это подход, при котором тесты используют разные наборы данных для проверки разных сценариев.