**Тестирование программного обеспечения** — проверка соответствия между реальным и ожидаемым поведением программы, осуществляемая на конечном наборе тестов, выбранном определенным образом. В более широком смысле, тестирование — это одна из техник контроля качества, включающая в себя активности по планированию работ (Test Management), проектированию тестов (Test Design), выполнению тестирования (Test Execution) и анализу полученных результатов (Test Analysis).  
  
**Качество программного обеспечения (Software Quality)** — это совокупность характеристик программного обеспечения, относящихся к его способности удовлетворять установленные и предполагаемые потребности. [Quality management and quality assurance]  
  
**Верификация (verification)** — это процесс оценки системы или её компонентов с целью определения удовлетворяют ли результаты текущего этапа разработки условиям, сформированным в начале этого этапа[IEEE]. Т.е. выполняются ли наши цели, сроки, задачи по разработке проекта, определенные в начале текущей фазы.  
**Валидация (validation)** — это определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, требованиям к системе [BS7925-1].  
Также можно встретить иную интерпритацию:  
Процесс оценки соответствия продукта явным требованиям (спецификациям) и есть верификация (verification), в то же время оценка соответствия продукта ожиданиям и требованиям пользователей — есть валидация (validation). Также часто можно встретить следующее определение этих понятий:  
Validation — ’is this the right specification?’.  
Verification — ’is the system correct to specification?’.  
  
**Цели тестирования**  
Повысить вероятность того, что приложение, предназначенное для тестирования, будет работать правильно при любых обстоятельствах.  
Повысить вероятность того, что приложение, предназначенное для тестирования, будет соответствовать всем описанным требованиям.  
Предоставление актуальной информации о состоянии продукта на данный момент.  
  
**Этапы тестирования:**  
1. Анализ продукта  
2. Работа с требованиями  
3. Разработка стратегии тестирования  
и планирование процедур контроля качества  
4. Создание тестовой документации  
5. Тестирование прототипа  
6. Основное тестирование  
7. Стабилизация  
8. Эксплуатация

**Тест план (Test Plan)** — это документ, описывающий весь объем работ по тестированию, начиная с описания объекта, стратегии, расписания, критериев начала и окончания тестирования, до необходимого в процессе работы оборудования, специальных знаний, а также оценки рисков с вариантами их разрешения.  
Отвечает на вопросы:  
Что надо тестировать?  
Что будете тестировать?  
Как будете тестировать?  
Когда будете тестировать?  
Критерии начала тестирования.  
Критерии окончания тестирования.   
  
**Тест дизайн** – это этап процесса тестирования ПО, на котором проектируются и создаются тестовые сценарии (тест кейсы), в соответствии с определёнными ранее критериями качества и целями тестирования.  
Роли, ответственные за тест дизайн:  
• Тест аналитик — определяет «ЧТО тестировать?»  
• Тест дизайнер — определяет «КАК тестировать?»

**Traceability matrix — Матрица соответствия требований** — это двумерная таблица, содержащая соответсвие функциональных требований (functional requirements) продукта и подготовленных тестовых сценариев (test cases). В заголовках колонок таблицы расположены требования, а в заголовках строк — тестовые сценарии. На пересечении — отметка, означающая, что требование текущей колонки покрыто тестовым сценарием текущей строки.  
Матрица соответсвия требований используется QA-инженерами для валидации покрытия продукта тестами. МСТ является неотъемлемой частью тест-плана.  
  
**Тестовый сценарий (Test Case)** — это артефакт, описывающий совокупность шагов, конкретных условий и параметров, необходимых для проверки реализации тестируемой функции или её части.  
Пример:  
Action Expected Result Test Result  
(passed/failed/blocked)  
Open page «login» Login page is opened Passed  
  
Каждый тест кейс должен иметь 3 части:  
PreConditions Список действий, которые приводят систему к состоянию пригодному для проведения основной проверки. Либо список условий, выполнение которых говорит о том, что система находится в пригодном для проведения основного теста состояния.  
Test Case Description Список действий, переводящих систему из одного состояния в другое, для получения результата, на основании которого можно сделать вывод о удовлетворении реализации, поставленным требованиям  
PostConditions Список действий, переводящих систему в первоначальное состояние (состояние до проведения теста — initial state)  
Виды Тестовых Сценариев:  
Тест кейсы разделяются по ожидаемому результату на позитивные и негативные:  
• Позитивный тест кейс использует только корректные данные и проверяет, что приложение правильно выполнило вызываемую функцию.  
• Негативный тест кейс оперирует как корректными так и некорректными данными (минимум 1 некорректный параметр) и ставит целью проверку исключительных ситуаций (срабатывание валидаторов), а также проверяет, что вызываемая приложением функция не выполняется при срабатывании валидатора.  
  
**Чек-лист (check list)** — это документ, описывающий что должно быть протестировано. При этом чек-лист может быть абсолютно разного уровня детализации. На сколько детальным будет чек-лист зависит от требований к отчетности, уровня знания продукта сотрудниками и сложности продукта.  
Как правило, чек-лист содержит только действия (шаги), без ожидаемого результата. Чек-лист менее формализован чем тестовый сценарий. Его уместно использовать тогда, когда тестовые сценарии будут избыточны. Также чек-лист ассоциируются с гибкими подходами в тестировании.  
  
**Дефект (он же баг)** – это несоответствие фактического результата выполнения программы ожидаемому результату. Дефекты обнаруживаются на этапе тестирования программного обеспечения (ПО), когда тестировщик проводит сравнение полученных результатов работы программы (компонента или дизайна) с ожидаемым результатом, описанным в спецификации требований.  
  
**Error** — ошибка пользователя, то есть он пытается использовать программу иным способом.  
Пример — вводит буквы в поля, где требуется вводить цифры (возраст, количество товара и т.п.).  
В качественной программе предусмотрены такие ситуации и выдаются сообщение об ошибке (error message), с красным крестиком которые.  
**Bug (defect)** — ошибка программиста (или дизайнера или ещё кого, кто принимает участие в разработке), то есть когда в программе, что-то идёт не так как планировалось и программа выходит из-под контроля. Например, когда никак не контроллируется ввод пользователя, в результате неверные данные вызывают краши или иные «радости» в работе программы. Либо внутри программа построена так, что изначально не соответствует тому, что от неё ожидается.  
**Failure** — сбой (причём не обязательно аппаратный) в работе компонента, всей программы или системы. То есть, существуют такие дефекты, которые приводят к сбоям (A defect caused the failure) и существуют такие, которые не приводят. UI-дефекты например. Но аппаратный сбой, никак не связанный с software, тоже является failure.  
  
**Баг Репорт (Bug Report)** — это документ, описывающий ситуацию или последовательность действий приведшую к некорректной работе объекта тестирования, с указанием причин и ожидаемого результата.  
Шапка  
Короткое описание (Summary) Короткое описание проблемы, явно указывающее на причину и тип ошибочной ситуации.  
Проект (Project) Название тестируемого проекта  
Компонент приложения (Component) Название части или функции тестируемого продукта  
Номер версии (Version) Версия на которой была найдена ошибка  
Серьезность (Severity) Наиболее распространена пятиуровневая система градации серьезности дефекта:  
• S1 Блокирующий (Blocker)  
• S2 Критический (Critical)  
• S3 Значительный (Major)  
• S4 Незначительный (Minor)  
• S5 Тривиальный (Trivial)  
Приоритет (Priority) Приоритет дефекта:  
• P1 Высокий (High)  
• P2 Средний (Medium)  
• P3 Низкий (Low)  
Статус (Status) Статус бага. Зависит от используемой процедуры и жизненного цикла бага (bug workflow and life cycle)  
  
Автор (Author) Создатель баг репорта  
Назначен на (Assigned To) Имя сотрудника, назначенного на решение проблемы  
Окружение  
ОС / Сервис Пак и т.д. / Браузера + версия /… Информация об окружении, на котором был найден баг: операционная система, сервис пак, для WEB тестирования — имя и версия браузера и т.д.  
…  
Описание  
Шаги воспроизведения (Steps to Reproduce) Шаги, по которым можно легко воспроизвести ситуацию, приведшую к ошибке.  
Фактический Результат (Result) Результат, полученный после прохождения шагов к воспроизведению  
Ожидаемый результат (Expected Result) Ожидаемый правильный результат  
Дополнения  
Прикрепленный файл (Attachment) Файл с логами, скриншот или любой другой документ, который может помочь прояснить причину ошибки или указать на способ решения проблемы  
  
**Severity vs Priority**  
Серьезность (Severity) — это атрибут, характеризующий влияние дефекта на работоспособность приложения.  
Приоритет (Priority) — это атрибут, указывающий на очередность выполнения задачи или устранения дефекта. Можно сказать, что это инструмент менеджера по планированию работ. Чем выше приоритет, тем быстрее нужно исправить дефект.  
Severity выставляется тестировщиком  
Priority – менеджером, тимлидом или заказчиком  
  
Градация Серьезности дефекта (Severity)  
  
**S1 Блокирующая (Blocker)**  
Блокирующая ошибка, приводящая приложение в нерабочее состояние, в результате которого дальнейшая работа с тестируемой системой или ее ключевыми функциями становится невозможна. Решение проблемы необходимо для дальнейшего функционирования системы.  
  
**S2 Критическая (Critical)**  
Критическая ошибка, неправильно работающая ключевая бизнес логика, дыра в системе безопасности, проблема, приведшая к временному падению сервера или приводящая в нерабочее состояние некоторую часть системы, без возможности решения проблемы, используя другие входные точки. Решение проблемы необходимо для дальнейшей работы с ключевыми функциями тестируемой системой.  
  
**S3 Значительная (Major)**  
Значительная ошибка, часть основной бизнес логики работает некорректно. Ошибка не критична или есть возможность для работы с тестируемой функцией, используя другие входные точки.  
  
**S4 Незначительная (Minor)**  
Незначительная ошибка, не нарушающая бизнес логику тестируемой части приложения, очевидная проблема пользовательского интерфейса.  
  
**S5 Тривиальная (Trivial)**  
Тривиальная ошибка, не касающаяся бизнес логики приложения, плохо воспроизводимая проблема, малозаметная посредствам пользовательского интерфейса, проблема сторонних библиотек или сервисов, проблема, не оказывающая никакого влияния на общее качество продукта.  
  
Градация Приоритета дефекта (Priority)  
**P1 Высокий (High)**  
Ошибка должна быть исправлена как можно быстрее, т.к. ее наличие является критической для проекта.  
**P2 Средний (Medium)**  
Ошибка должна быть исправлена, ее наличие не является критичной, но требует обязательного решения.  
**P3 Низкий (Low)**  
Ошибка должна быть исправлена, ее наличие не является критичной, и не требует срочного решения.