1. Что подразумевается под тестовой средой?
2. В чем отличие build от release?
3. Основные фазы тестирования?
4. Что такое импакт анализ?
5. Критерии выбора тестов?
6. Принципы тестирования?
7. Что такое независимое тестирование?
8. Что такое тестирование со сдвигом влево?
9. Почему тестирование делится на отдельные этапы?
10. Какие есть этапы тестирования?
11. Как вы тестируете продукт, если требования еще не зафиксированы?
12. Как вы узнаете, было ли создано достаточно тестов для тестирования продукта?
13. Что подразумевается под тестовыми данными?
14. Что такое бета-тестирование?
15. Что означает пилотное тестирование? (Pilot)
16. В чем отличие build от release?
17. Что такое бизнес – логика?
18. Какие существуют основные виды тестирования ПО?
19. Какие существуют основные виды тестирования ПО?
20. Методы тестирования?
21. Что такое пирамида / уровни тестирования?
22. Что такое деструктивное/разрушающее/негативное тестирование? (DT — Destructive testing)
23. Что такое недеструктивное/неразрушающее/позитивное тестирование? (NDT – Non Destructive testing)
24. Что подразумевается под компонентным/модульным/юнит тестированием? (Component/Module/Unit testing)
25. Что подразумевается под интеграционным тестированием?

1.Что подразумевается под тестовой средой?

**Среда тестирования** — это настройка программного и аппаратного обеспечения для групп тестирования для выполнения тестовых случаев. Другими словами, он поддерживает выполнение теста с настроенным оборудованием, программным обеспечением и сетью.

**A test environment** is a software and hardware setup for test teams to run test cases. In other words, it supports test execution with configured hardware, software, and network.

**Testiympäristö on ohjelmisto-** ja laitteistoasetus testausryhmille testitapausten suorittamista varten. Toisin sanoen se tukee testin suorittamista määritetyillä laitteilla, ohjelmistoilla ja verkossa.

**Środowisko testowe** to konfiguracja oprogramowania i sprzętu dla zespołów testowych do uruchamiania przypadków testowych. Innymi słowy, obsługuje wykonywanie testów ze skonfigurowanym sprzętem, oprogramowaniem i siecią.

2. В чем отличие build от release?

Build. A development activity whereby a complete system is compiled and linked, so that a consistent system is available including all latest changes. [На основе определения термина «daily build» из ISTQB Glossary]

**Отличие** между сборкой и выпуском в тестировании программного обеспечения является то, что **Build - это версия программного обеспечения, которую команда разработчиков передает команде тестирования для целей тестирования, а Release - это программное обеспечение, которое команда тестирования передает клиенту.**

**3.** Основные фазы тестирования?

1. Создание тестового набора (test suite) путем ручной разработки или автоматической генерации для конкретной среды тестирования (testing environment).
2. Прогон программы на тестах, управляемый тестовым монитором (test monitor, test driver [IEEE Std 829-1983]) с получением протокола результатов тестирования (test log).
3. Оценка результатов выполнения программы на наборе тестов с целью принятия решения о продолжении или остановке тестирования.

**4.Что такое импакт анализ?**

Импакт анализ — еще один инструмент в разработке ПО, который применяется программистами для ~~облегчения жизни тестировщиков~~ улучшения тестирования. сновная идея импакт анализа — это **указание затронутых мест в проекте при разработке новой или изменении старой функциональности.** Во время разработки часто возникают ситуации, когда вносишь изменения в одном месте, а это влияет либо на весь проект либо на несколько мест в проекте. При таких изменениях сложно вспомнить и протестировать все необходимые места, особенно если они не покрыты тестами. В результате этого, после сдачи очередного релиза, клиенты начинают обращаться с жалобами на то, что что-то поломалось или не работает.

**5.Критерии выбора тестов?**

1. **достаточность**, т. е. критерий должен показывать, когда некоторое конечное множество тестов достаточно для тестирования данной программы;

2. **полнота**, т. е. в случае ошибки должен существовать тест из множества тестов, удовлетворяющих критерию, который раскрывает ошибку;

3. **надежность,** т. е. любые два множества тестов, удовлетворяющих ему, одновременно должны раскрывать или не раскрывать ошибки программы;

4***. проверяемость,*** т. е. критерий должен быть легко проверяемым, например, вычисляемым на тестах.

**простая логика:** •

Тесты ищут ошибки. •

Но все ошибки найти невозможно.

• Значит, наша задача — найти максимум ВАЖНЫХ ошибок за имеющееся время.

**Приступая к продумыванию чек-листа, тест-кейса или набора тест-кейсов, задайте себе следующие вопросы и получите чёткие ответы**

Что перед вами? Если вы не понимаете, что вам предстоит тестировать, вы не уйдёте дальше бездумных формальных проверок.

• Кому и зачем оно нужно (и насколько это важно)? Ответ на этот вопрос позволит вам быстро придумать несколько характерных сценариев использования того, что вы собираетесь тестировать.

• Как оно обычно используется? Это уже детализация сценариев и источник идей для позитивного тестирования{79} (их удобно оформить в виде чек-листа).

• Как оно может сломаться, т.е. начать работать неверно? Это также детализация сценариев использования, но уже в контексте негативного тестирования (их тоже удобно оформить в виде чек-листа).

**6.Принципы тестирования?**

Принцип 1. Тестирование показывает наличие дефектов  
Тестирование может показать, что дефекты присутствуют, но не может доказать, что дефектов нет

Принцип 2. Исчерпывающее тестирование невозможно

Принцип 3. Раннее тестирование

Принцип 4. Скопление дефектов

Принцип 5. Парадокс пестицидов

Принцип 6. Тестирование зависит от контекста

Принцип 7. Заблуждение об отсутствии ошибок

Principle 1. Testing shows defects

Principle 2: Exhaustive testing is not possible

Principle 3. Early testing

Principle 4. Accumulation of defects

Principle 5. The pesticide paradox

Principle 6. Testing is context sensitive

Principle 7. The error-free fallacy

**7.Что такое независимое тестирование?**

**Независимое тестирование** (аутсорсинг **тестирования**) - это **тестирование** силами дополнительно привлечённой компании или группы людей, **не участвующих непосредственно в разработке программного обеспечения.**

**8.Что такое тестирование со сдвигом влево?**

Принцип «Shift left» привлекает команду тестировщиков еще на ранней стадии разработки программного обеспечения. Следовательно, они могут четко понимать требования, архитектуру ПО и разрабатывать тестовые примеры уже на начальном этапе, появляется возможность исправлять все сбои как можно раньше.

Это понимание позволяет тестировщикам получить полные знания о продукте, проанализировать и спроектировать различные сценарии, которые помогут команде выявить дефекты даже до того, как будет выполнено написание кода.

**Смысл концепции «Shift Left» — это найти дефекты как можно раньше всеми возможными способами.**

**9.Почему тестирование делится на отдельные этапы?**

**10.Какие есть этапы тестирования?**

Следуя общей логике итеративности, превалирующей во всех современных

моделях разработки ПО, жизненный цикл тестирования также выражается замкнутой последовательностью действий (рисунок 2.1.g)(общее планирование и анализ, уточнение критериев приемки, уточнение стратегии тестирования, разработка тесткейсов, выполнение тест-кейсов, фиксация найденных дефектов, анализ результатов тестирования, отчетность), направленных на обеспечение качества ПО

11.**Как вы тестируете продукт, если требования еще не зафиксированы?**

Ad hoc

12. Как вы узнаете, было ли создано достаточно тестов для тестирования продукта?

Прежде всего, мы проверим, охватывает ли каждое требование хотя бы один Test case. Если да, то можно сказать, что тестовых примеров достаточно для тестирования продукта. *(test case) — набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства*

**13. Что подразумевается под тестовыми данными?**

Наиболее ресурсоёмкая часть тестирования - это подготовка тестовых данных. Они зависят от типа и цели тестирования, стадии разработки проекта и много от чего ещё.

Тестовые данные - это набор входных значений, необходимых для выполнения Test case. тестировщики определяют данные в соответствии с требованиями. Они могут сделать это вручную или использовать инструменты генерации.

1. **Что такое бета-тестирование?** • **Бета-тестирование (beta testing)** выполняется вне организации-разработчика с активным привлечением конечных пользователей/заказчиков. Может являться формой внешнего приёмочного тестирования . Суть этого вида вкратце: продукт уже можно открыто показывать внешним пользователям, он уже достаточно стабилен, но проблемы всё ещё могут быть, и для их выявления нужна обратная связь от реальных пользователей.

**Beta testing.** Operational testing by potential and/or existing users/customers at an external site not otherwise involved with the developers, to determine whether or not a component or system satisfies the user/customer needs and fits within the business processes. Beta testing is often employed as a form of external acceptance testing for off-the-shelf software in order to acquire feedback from the market. [ISTQB Glossary]

**Beetatestaus. Potentiaalisten** ja / tai olemassa olevien käyttäjien / asiakkaiden suorittama operatiivinen testaus ulkoisella sivustolla, joka ei muuten ole mukana kehittäjissä, sen selvittämiseksi, täyttääkö komponentti tai järjestelmä käyttäjän / asiakkaan tarpeet ja sopiiko se liiketoimintaprosesseihin. Beetatestausta käytetään usein ulkoisten ohjelmistojen ulkoisena hyväksyntätestinä palautteen saamiseksi markkinoilta. [ISTQB-sanasto]

**Testowanie beta.** Testowanie operacyjne przez potencjalnych i / lub istniejących użytkowników / klientów w zewnętrznej lokalizacji, która nie jest w inny sposób zaangażowana w programistów, w celu ustalenia, czy komponent lub system spełnia potrzeby użytkownika / klienta i pasuje do procesów biznesowych. Testy beta są często wykorzystywane jako forma zewnętrznych testów akceptacyjnych dla gotowego oprogramowania w celu uzyskania informacji zwrotnej z rynku. [Glosariusz ISTQB]

15.Что такое пилотное тестирование?

PILOT testing определяется как тип тестирования программного обеспечения, который проверяет компонент системы или всю систему **в режиме реального времени. Целью пилотного теста является оценка осуществимости,** **времени, стоимости, риска и эффективности исследовательского проекта**. Это тестирование проводится точно между UAT и Production. В пилотном тестировании выбранная группа конечных пользователей пробует тестируемую систему и предоставляет обратную связь до полного развертывания системы. Другими словами, это означает проведение генеральной репетиции для последующего теста на удобство использования. Пилотное тестирование помогает в раннем обнаружении ошибок в Системе.

Пилотное тестирование связано с установкой системы на площадке заказчика (или в среде, моделируемой пользователем) для тестирования на предмет постоянного и регулярного использования. Выявленные недостатки затем отправляются команде разработчиков в виде отчетов об ошибках, и эти ошибки исправляются в следующей сборке системы. Во время этого процесса иногда приемочное тестирование также включается как часть тестирования на совместимость. Это происходит, когда система разрабатывается для замены старой.

**16.Что такое бизнес – логика?**

**Бизнес – логика**  это то, что конкретная программа по задумке должна сделать. Например, в складской программе проверка на возможность отправить товар (вдруг его нет в наличии). Это правила, которые должны соблюдаться в данной конкретной программе, определенные бизнес-клиентом. Слои приложения – слой пользовательского интерфейса, слой бизнес логики, слой сохранения данных.

**18.Виды тестирования**

* **Функциональные виды («Что?» - проверяет весь функционал продукта**):
* Функциональное тестирование (Functional testing)
* Тестирование взаимодействия (Interoperability testing)
* **Нефункциональное («Как?»):**
* **Производительности (**Performance)
* Тестирование емкости/способностей (Capacity testing)
* Стрессовое (Stress testing)
* Нагрузочное (Load testing)
* Объемное тестирование (Volume testing)
* Выносливости (Soak/Endurance testing)
* Стабильности/надежности (Stability / Reliability testing)
* Шиповое (Spike)
* Отказоустойчивости (Stability testing)
* Масштабируемости (Scalability test)
* Отказ и восстановление (Failover and Recovery testing)
* Удобство пользования (Usability testing)
* Тестирование установки (Installation testing)
* Тестирование безопасности (Security and Access Control testing)
* Конфигурационное (Configuration testing)
* **Связанное с изменениями:**
* Регрессионное (Regression testing)
* Санитарное (Sanity testing)
* Дымовое (Smoke testing)
* Тестирование сборки (Build Verification testing)

В разных источниках предлагается разный взгляд:

**20 Методы тестирования?**

Самым высоким уровнем в иерархии подходов к тестированию будет понятие типа, которое может охватывать сразу несколько смежных техник тестирования. То есть, одному типу тестирования может соответствовать несколько его видов. Отличаются они знанием внутреннего устройства объекта тестирования.

**White/grey/black box**

**21.Что такое пирамида / уровни тестирования?**

«Пирамида тестов» — метафора, которая означает группировку тестов программного обеспечения по разным уровням детализации. Она также дает представление, какое количество тестов должно быть в каждой из этих групп.

**22.Что такое деструктивное/разрушающее/негативное тестирование? (DT — Destructive testing)**

ОТРИЦАТЕЛЬНОЕ ТЕСТИРОВАНИЕ - тип тестирования ПО для поиска точек отказа в программном обеспечении, который проверяет систему на обработку исключительных ситуаций (срабатывание валидаторов на некорректные данные), а также проверяет, что вызываемая приложением функция не выполняется при срабатывании валидатора. Неожиданные условия могут быть чем угодно, от неправильного типа данных до хакерской атаки. Целью отрицательного тестирования является предотвращение сбоя приложений из-за некорректных входных данных. Просто проводя положительное тестирование, мы можем только убедиться, что наша система работает в нормальных условиях. Мы должны убедиться, что наша система может справиться с непредвиденными условиями, чтобы обеспечить 100% безошибочную систему.

*Типичные примеры: ввести неправильно составленный e-mail и номер телефона, загрузить файл не предусмотренного расширения или размера.*

Для деструктивного тестирования существует множество способов его тестирования:

Метод анализа точек отказа: это пошаговое прохождение системы, проводящее оценку того, что может пойти не так в разных точках. Для этой стратегии может быть использована помощь BA (Business Analyst).

Экспертная проверка тестировщика: проанализируйте или дайте на ревью ваши Test вашему коллеге-тестировщику, который менее знаком с системой/функцией

Бизнес-анализ тестовых случаев. Конечные пользователи или эксперты могут подумать о многих допустимых сценариях, которые иногда тестировщики могут не учитывать или упустить, так как все их внимание будет сосредоточено на тестировании требований.

Проведите предварительное тестирование с использованием контрольных таблиц (run sheets). Исследовательское тестирование с использованием контрольных таблиц поможет определить, что было проверено, повторить тесты и позволит вам контролировать охват тестами.

Используйте другой источник: вы можете попросить кого-нибудь сломать программный продукт и проанализировать различные сценарии.

**22.Что такое недеструктивное/неразрушающее/позитивное тестирование? (NDT – Non Destructive testing)**

Позитивное тестирование (positive testing) направлено на исследование приложения в ситуации, когда все действия выполняются строго по инструкции без каких бы то ни было ошибок, отклонений, ввода неверных данных и т.д. Если позитивные тест-кейсы завершаются ошибками, это тревожный признак — приложение работает неверно даже в идеальных условиях (и можно предположить, что в неидеальных условиях оно работает ещё хуже). Для ускорения тестирования несколько позитивных тест-кейсов можно объединять (например, перед отправкой заполнить все поля формы верными значениями) — иногда это может усложнить диагностику ошибки, но существенная экономия времени компенсирует этот риск.

**23.Что подразумевается под компонентным/модульным/юнит тестированием? (Component/Module/Unit testing)**

Модульное тестирование (юнит-тестирование). Модульные тесты используются для тестирования какого-либо одного логически выделенного и изолированного элемента системы (отдельные методы класса или простая функция, subprograms, subroutines, классы или процедуры) в коде. Очевидно, что это тестирование методом белого ящика и чаще всего оно проводится самими разработчиками.

|  |  |
| --- | --- |
| Unit testing | Component testing |
| Тестирование отдельных программ, модулей, функций для демонстрации того, что программа выполняется согласно спецификации | Тестирование каждого объекта или частей программного обеспечения отдельно с или без изоляции других объектов |
| Проверка в(на) соответствии с design documents | Проверка в(на) соответствии с test requirements, use case |
| Пишутся и выполняются(обычно) разработчиками | Тестировщиками |
| Выполняется первым | Выполняется после Unit |

Другой источник:

Разница между компонентным и модульным тестированием: По-существу эти уровни тестирования представляют одно и тоже, разница лишь в том, что в компонентном тестировании в качестве параметров функций используют реальные объекты и драйверы, а в модульном тестировании - конкретные значения.