**1). Что подразумевается под тестовой средой?**

**Тестовая среда (test environment)-** Программное и аппаратное обеспечение необходимое для проведения теста. Это окружение, в котором работает команда QA.

**2.) В чем отличие build от release?**

**Build**- продукт для программистов, тестировщиков.

**Release –**тот же продукт, но протестированный, с исправленными багами, готовый для использования заказчиком, пользователем.

**3.) Основные уровни тестирования?**

1. **Модульное тестирование.** Тестирование отдельных компонентов
2. **Интеграционное тестирование.** Проверка того как отдельные компоненты взаимодействуют между собой
3. **Системное тестирование.** Системное тестирование охватывает всю систему .целиком. Например, как телеграмм взаимодействует с другими приложениями.

**4) Что такое импакт анализ?**

**Импакт анализ** инструмент, который используется в разработке ПО программистом для улучшения тестирования. Заключается в том, что программист отмечает затронутые места при разработке новой или изменённой функциональности.

**5) Критерии выбора тестов?**

**- критерий должен быть достаточным – критерий не** показывает какое количество тестов необходимо для тестирования.

**-полным-** т.е. любые два множества тестов, удовлетворяющих ему, одновременно должны раскрывать или не раскрывать ошибки программы.

**-надежным** - т.е. любые два множества тестов, удовлетворяющих ему, одновременно должны раскрывать или не раскрывать ошибки программы.

**-легко проверяемым**, например вычисляемым на тестах

**6) Принципы тестирования**

1. **тестирование демонтирует наличие дефектов-** тестирование может показать, что дефекты есть, но не может доказать, что никаких дефектов не существует.
2. **исчерпывающее тестирование невозможно-** протестировать все невозможно, за исключение тривиальных случаев
3. **раннее тестирование-** тесты должны начинаться как можно раньше в процессе разработки
4. **скопление дефектов** небольшое количество модулей содержат большинство дефектов, выявленных в ходе тестирования. Принцип Парето 80% ошибок содержится в 20% функций.
5. **парадокс пестицида** если тестировать одними и теми же тестами, то они перестанут находить дефекты.
6. **тестирование зависит от контекста тестирование** выбор техники, типа тестирования, во многом зависит от того какой продукт мы тестируем. Например, ПО для медицинских нужд требует более строгой проверки чем компьютерная игра.
7. **заблуждение об отсутствии ошибок** если тестирование не обнаружило ошибок, это не значит, что продукт готов к релизу. Найденные дефекты будут не важны, если система окажется не удобной.

**7) Что такое независимое тестирование?**

Это тестирование, которое выполняется не разработчиками, а другим отделом или организацией

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

Это такой подход к тестированию ПО, при котором тестирование выполняется на более ранней стадии. Т.е. перемещается влево по временной шкале проекта.

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

-каждый этап имеет свое значение

-проще управлять поэтапно

- можем запустить разные тесты в разных средах

-производительность и качество улучшается за счет поэтапного тестирования

10. Какие есть этапы тестирования?

**1. Анализ требований.** Сбор требований к разрабатываемому ПО, документирвание, анализ. Бизнес-аналитики уточняют требования у PO и документируют их

**2. Проектирование.** На этой стадии программисты и системные архитекторы разрабатывают дизайн системы согласно требований QA тест-план, тестксейс,.

**3. Разработка** Во время процесса следует провести модульное, интеграционное и системное тестирование.

**4. Тестирование** На этом этапе находят и устраняют ошибки.

**5 Эксплуатация поддержка**

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

Если спецификация требований недоступна для продукта, тогда план тестирования может быть создан на основе предположений, сделанных относительно продукта. Но мы должны хорошо документировать все предположения в плане тестирования.

## 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.

## 13. Что подразумевается под тестовыми данными?

Это набор входных значений, который необходим для выполнения test case. Эти данные мы берем из требований. Можно делать это вручную или при помощи инструментов генерации

## 14. Что такое бета тестирование

Это тестирование, которое проводится клиентами или конечными пользователями. При бета тестирования используется Black Box Testing. Не требует какой-либо среды тестирования. Занимает несколько недель. Большинство вопросов и собранных в результате тестирования будет реализованы в следующих версиях.

* **Традиционное бета тестирование**: продукт распространяются на целевой рынок и данные собираются по всем аспектам.
* **Публичное бета тестирование:** продукт публикуется во внешнем мире через –онлайн каналы и данные могут быть получены от любого пользователя.
* **Техническое бета тестирование** продукт передается во внутреннюю группу организации и собирает отзывы от сотрудников организации.
* **Целевая бета версия** продукт выпущен на рынок для сбора отзывов об особенностях программы
* **Бета версия после выпуска .** Продукт выпущен на рынок, и данные собираются для внесения улучшений в будущем выпуске продукта.

## 15. Что означает пилотное тестирование? (Pilot)

Это тип тестирования, который проверяет всю систему или отдельные компоненты в режиме реального времени. Это тестирование проводится между UAT (приемочное пользовательское User Acceptance Testing) и Production. Берется группа пользователей и пробует тестирует систему и дает обратную связь об использовании продукта.

## 17. Что такое бизнес – логика?

Это, то что по задумке должна выполнять программа.

Двухслойная архитектура- на клиенте находится пользовательский интерфейс и часть бизнес логики, на сервере- часть бизнес логики и бд.

Трехуровневая архитектура. Каждый слой располагается отдельно (слой логики представления, слой бизнес логики, слой данных).

Задача бизнес логики брать данные с нижнего слоя (базы данных) в соответствии запросом пользователя пришедшим с верхнего слоя (веб браузера).

## 18. Какие существуют основные виды тестирования ПО?

1. **Функциональные виды тестирования**- проверяют весь функционал продукта

- ***Функциональное тестирование***- проверяют весь функционал продукта.

-***Тестирование взаимодействия***-тестирование, которое проверяет способность приложения взаимодействовать соседним и более компонентом или системами.

- ***Тестирование безопасности***-метод тестирования, при котором проверяется защищает ли система данные.

**2. Нефункциональные виды тестирования –**Это тестирование того, как работает система. (быстро, надежно, безопасно)

**-тестирование производительности**  (Performance)

**- тестирование установки (**Installation testing)

**- тестирования удобства использования (**Usability Testing)

- **тестирование на отказ и восстановление**

- **конфигурационное тестирование** (Configuration Testing)

-тестирование безопасности

**3. Связанное с изменениями:**

- Регрессионное тестирование  (Regression testing)

-Санитарное тестирование  (Sanity testing)

- Smoke тестирование (Smoke testing)

-Тестирование сборки (Build Verification testing)

## 19. Методы тестирования

**Метод белого ящика-** пред полает, что тестировщик знает как устроена система внутри. На основании этого он выбирает входные значения для проведения тестирования и знает, какой ответ должна выдать система. Преимущества этого метода заключается в том, что тестирование может проводится на раннем этапе и не нужно ждать пользовательского интерфейса. Недостатком такого метода тестирования является, то что для него необходимы более глубокие знания.

Основным методом тестирования является

- анализ покрытия операторов-все ли операторы били проверены набором тестов

-анализ покрытия условий- все ли пути покрытия кодов были проверены набором тестов

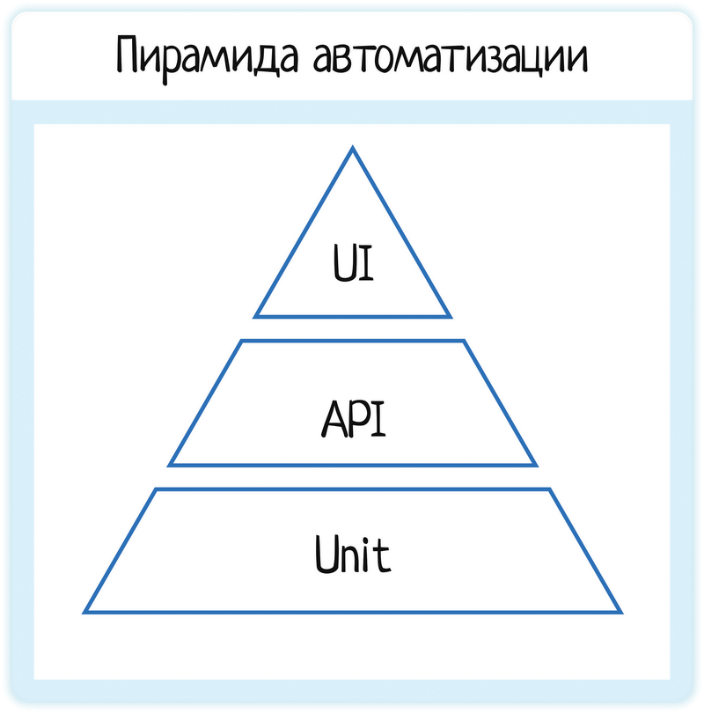
-анализ покрытия решений

**Метод серого ящика-** метод предполагает комбинацию Белого и Серого ящика.т.е внутренне устройство программы известно лишь частично. Т.е. он знает как устроено приложение, но не видит код самого приложения.

**Метод черного ящика метод** при котором мы не знаем как программа устроена внутри, данный метод работает исключительно с внешним интерфейсом.

## 20.Что такое пирамида / уровни тестирования? (Testing Levels)

Пирамида тестов- группировка тестов ПО по уровням детализации. Она так же дает понять сколько тестов должно быть в каждой из групп.



Ui -тесты на отдельную мелкую функцию (посчитать одну ячейку отчета)

API- тесты на конкретный функционал, который состоит из отдельных функций. (загрузить весь отчет)

Unit- честный тест через графический интерфейс «как это делал бы пользователь»(открыл браузер, вошел в систему, перешел в отчеты, вызывал отчет)

## 21. Что такое деструктивное/разрушающее/негативное тестирование? (DT — Destructive testing)

Это тестирование, которое проводится после того как проведено позитивное тестирование и мы располагаем временем на его проведение. Т.е. негативное тестирование проверяет систему на ввод невалидных данных, исключительных ситуаций. Главная задача этого метода тестирования заключается в предотвращении сбоя в продукте при вводе некорректных входных данных. (ввести неправильный номер, загрузить файл с непредусмотренным расширением )

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

Позитивное тестирование показывает, что наш продукт работает как ожидается. Когда ПП отработает корректно на правильных значениях.

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

Модульное/юнит тестирование — это вид тестирования, который используется для логически выделенного, изолированного элемента системы. Это тестирование проводится методом белого ящика и чаще всего самим разработчиком. На этом уровне легче всего проверить ошибки кодирования алгоритмов, а допустим ошибки в трактовке данных, некорректной реализацией интерфейса обычно пропускаются на этом уровне.

## 24. Что подразумевается под интеграционным тестированием?

**Интеграционное тестирование** - тестирование, при котором проверяют как две и более части приложения взаимодействуют между собой. Для интеграционного тестирования уже используются компоненты проверенный с помощью модульного тестирования.

Уровни интеграционного тестирования

-**компонентный интеграционный**- проверяет взаимодействие между компонентами системы, после проведения компонентного тестирования.

**-системный интеграционный –** проверяет взаимодействие между разными системами после проведения системного тестирования

## 25.Разница между Unit testing и Integration testing?

Если тест использует базу данных, использует сеть для вызова другого компонента, читает записывает файлы, то это **интеграционный тест**, а не модульный.

Теоретически можно использовать только интеграционные тесты, но на практике:

-юнит-тесты легче поддерживать

-юнит-тесты быстрее интеграционных

-сбойные юнит тесты легче исправить

Нужно использовать и интеграционные и модульные тесты т.к. они дополняют друг груда, а не заменяют.

## 26.Что такое системное интеграционное тестирование? (SIT — System Itegration testing)

В этом типе тестирования модули сначала тестируются индивидуально, а затем объединяются в систему и тестируются постепенно, до тех пор, пока не будет интегрирована вся система.

## 27. В чем разница между тест-драйвером и тест-заглушкой? (Test Driver and Test Stub)

## Тестовый драйвер - это фрагмент кода, который вызывает тестируемый программный компонент. Это полезно при тестировании по принципу «снизу-вверх».

## Тестовая заглушка - это фиктивная программа, которая интегрируется с приложением для полной функциональности. Они актуальны для тестирования, в котором используется нисходящий подход.

1. Что подразумевается под системным тестированием?

## Системное тестирование рассматривает систему в целом и производится с помощью черного ящика. Основная задача тестирования проверить как система работает в целом. Для минимизации рисков рекомендуются использовать окружение мах приближенное к тому, на котором будет установлен продукт после выдачи

## Методы HTPP запросов.

**Get**- отправляет собранную информацию как часть URL 2056 символов. Данные кешируются

http://www.komtet.ru/script.php?login=admin&name=komtet

**Post** – передает информацию, таким образом, что пользователь ее не видит. У поста есть тело запроса и не кешируются данные. Этим методом можно передавать файлы.

**PUT-** обновляет ресурс

**Delite**- удаляет ресурс

**OPTIONS**- используется для описания параметров

**Connect**-устанавливает тоннель к серверу

**Коды ответов/состояния сервера с примерами? (HTTP status codes)**

* Информационные (100-105) 100-продолжить, 102- в обработке, 103-ранняя подсказка
* Успешные (200-226) 200- ок, 202-принято, 204-нет сожержимого
* Перенаправление (300-307) 301- перенаправление на постоянной основе
* Ошибка клиента (400-499) 400-плохой запрос, 404-не найден, 401-не авторизовано, аутентификация возможна
* Ошибка сервера (500-510) 500- внутренняя ошибка сервера,

**Жизненный цикл продукта**

**Анализ требований –** уточняются требования к продукту и документируются

* Участники: Product owner, BA(бизнес-аналитик), QA.
* Артефакты: спецификация требований

**Проектирование-** разработка дизайна системы

* Участники: Product owner, разработчики, системные архитекторы, QA.
* Артефакты: дизайн-спецификация (Design Specification Document, DSD), Тест-план, тест-сценарии, тест-кейсы; вариативно: тестовая стратегия.

**Разработка-** настройка окружения, разработка польз интерфейса, разработчики пишут unit тесты

* **Участники:** разработчики

**Тестирование** Участники: QA. Артефакты: дефект-репорты

**Внедрение и сопровождение**.

* Участники: команда технической поддержки. Артефакты: замечания, запросы на исправление/улучшение.