Теория тестирования
1. Почему практики тест-дизайна нельзя применять сразу после получения требований?
Требования могут быть неидеальны - или неполными, или с ошибками. Поэтому сначала надо требования проанализировать - декомпозировать их на составляющие (слона легче съесть по кускам), визуализировать для наглядности, провести поиск серых зон.
2. В какой ситуации классы эквивалентности и граничные значения могут существовать по отдельности? Аргументируй свой ответ и приведи примеры.
Если класс эквивалентности представлен в виде набора значений, то у него не может быть граничных значений. Набор представляет собой ограниченное множество прописанных значений. Для примера: варианты выбора в выпадающем списке - набор, количество элементов которого должно быть описано в требованиях. Граничных значений в этом примере нет.
3. Что такое эквивалентность? Что такое класс эквивалентности?
Эквивалентность - равноценность некоторых объектов. Класс эквивалентности - некое множество объектов, которые будут обрабатываться одинаково и результат этой обработки также будет одинаковым.
4. Можно ли исключить проверку в середине диапазона в пользу проверок на границах, входящих в диапазон? Аргументируй
Граничные значения - места, где часто возникают ошибки. Если исключить проверку в середине диапазона, то может возникнуть ложное убеждение о неправильной работе ПО на всём диапазоне, тогда как ошибки были только в ГЗ, а в середине диапазона их не было, или же наоборот.
5. Представь, что тебе нужно протестировать форму: у каждого поля есть валидатор. Результат работы формы зависит от комбинации данных в полях. Какие практики тест-дизайна следует применить и почему? Аргументируй свой ответ.
Следует применить позитивные проверки (убедиться, что форма работает как надо при вводе корректных данных), негативные проверки (убедиться, что форма корректно реагирует на некорректные введенные данные), класс эквивалентности (проверки количества символов в поля и их сочетаний) и граничные значения (проверки граничных значений выделенных диапазонов), таблица принятия решения (для максимального покрытия проверками).
6. Какими способами можно оптимизировать количество проверок при работе с таблицами принятия решений? Аргументируй
- удаление дублирующихся проверок (экономия сил и времени); - удаление проверок с взаимоисключаемыми параметрами (к примеру мужчина не может быть одновременно холост и женат) - применить технику попарного тестирования (если пара параметров вызвала ошибку в комбинации с одеими параметрами, с большой вероятностью та же пара вызовет ошибку и в другой комбинации, поэтому достаточно протестировать только уникальные пары вариантов)
7. Опиши, чем чек-лист отличается от тест-кейсов. Приведи примеры, где применяют и то, и другое.
Чек-лист, в отличии от тест-кейсов, не содержит шагов теста и (чаще всего) ожидаемого результата, т.е. чек-лист предельно лаконичен. Также в отличии от чек-листа тест-кейс является проверкой только одного элемента. Применение и тест-кейсов и чек-листов возможно в случае нехватки времени на покрытие тест-кейсами всего функционала. В этом случае можно тест-кейсами покрыть наиболее важную часть функционала, остальное же - чек-листами.
8. Как правильно составить баг-репорт? Какие элементы баг-репорта — обязательные? Почему?

- для начала убедиться в воспроизводимости бага и в том, что репорт на него не был заведен ранее;
- коротко, но ёмко описатьт суть бага в заголовке (заголовок должен отвечать на вопросы "Что? Где? Когда?";
- описать шаги воспроизведения - алгоритм воспроизведения бага;
- описать ожидаемый результат (можно сослаться на соответствующий пункт в требованиях);
- описать фактический результат;
- указать окружение (модель устройства, ОС, версии браузера или приложения);
- по возможности приложить скриншоты, логи, файлы для воспроизведения багов;
- указать приоритет бага (его критичность и срочность исправления)

Обязательные элементы баг-репорта:
- ID - уникальный номер бага (необходим для удобного хранения в базе данных и последующего быстрого поиска);
- заголовок (позволяет быстро понять суть бага без перечитывания шагов);
- шаги воспроизведения (необходимая инструкция для восроизведения бага);
- результаты (ОР и ФР) (необходим для экономии времени поиска данной информации и для наглядного сравнения требований с фактическим результатом);
- окружение (отсутствие этой информации повлечет потерю уймы времени на поиск нужного окружения);
- приоритет (необходим для приоритизации багов и выделения некоторых из них для скорейшего исправления)

9. По каким правилам составляют заголовок баг-репорта? Что будет, если составить заголовок неправильно?

Заголовок должен предельно точно и коротко отвечать на вопросы "Что? Где? Когда?"
Что? - Что произошло во время бага?
Где? - Где и в каком месте приложения был замачен баг?
Когда? - При каких действиях и условиях произошел баг?

Неправильное составление заголовка может привести к:
- потере времени на обработку бага при расхождении информации в заголовке и шагах или неполной информации в заголовке;
- потере времени на поиск определенного бага

10. Из чего состоит клиент-серверная архитектура приложения? Кратко опиши функциональность каждого элемента.

Клент-серверная архитектура состоит из:
- клиента - система, кторая связывается с сервером и запрашивает у него нужную пользователю информацию;
- сервера - система, которая обрабатывает запросы клиента и отправляет обратно ответ;
- сеть (интернет) - система, которая помогает обмениваться информацией между клиентом и сервером.

11. Опиши этапы обработки запроса после того, как в адресную строку браузера вводят URL: https://yandex.ru.

- браузер после ввода URL отправляет запрос к DNS-серверу;
- DNS-сервер возвращает браузеру нужный IP-адрес сервера;
- браузер оправляет запрос серверу по полученному IP-адресу;
- сервер обрабатывает запрос и отправляет клиенту ответ с нужной информацией.

12. Что такое кэш? Зачем он нужен? Какое правило нужно соблюдать при работе с кэшем в тестировании?

Кэш - данные веб страниц, которые хранятся локально на компьютере. Кэш нужен для ускореня загрузки веб страниц, т.к. часть информации подгружается прямо из памяти ПК без подключения к сети. При работе с кэшем в тестировании надо помнить - перед тестированием исправленных или новых версий веб страниц необходимо почистить кэш браузера. В кэше может содержаться устаревшая версия веб страницы и она может загрузится, тем самым отобразив уже исправленные в новой версии дефекты.

13. Ты знаешь, что есть протоколы HTTP и HTTPS.
- Чем отличаются HTTP и HTTPS? В каких случаях не стоит пользоваться HTTP?
- Из каких компонентов состоит HTTP запрос: за что отвечает каждый?
- Какие HTTP-методы ты знаешь? За что они отвечают? Приведи примеры применения разных методов.
- Что такое код HTTP-ответа? Какие коды бывают?

HTTPS отличается от HTTP наличием шифрования соединения по криптографическим протоколам. В случае, когда на сайте планируется сбор личной информации (ФИО, адрес, номера банковских карт и прочее), пользование протоколом HTTP крайне не рекомендуется, т.к. это может привети к краже этой информации.

HTTP запрос состоит из следующих компонентов:
- стартовая строка (содежит метод запроса, путь до нужного ресурса, версию используемого протокола);
- заголовок (содежит информацию о клиенте и параметры запроса);
- тело запроса (данные, которые передает клиент серверу).

HTTP-методы:
- GET - запрос данных у бэкенда у определенного ресурса (с помощью GET запроса можно подключиться к нужным веб страницам, получить нужную информацию);
- POST - отправка данных на бэкенд (с помощью POST запроса можно произвести загрузку различных файлов на сервер, отправить данные на сервер (например, данные пользователя для регистрации));
- PUT - изменение имеющихся данных на бэкенде (с помощью PUT запроса можно изменть уже имеющиеся данные - например, исправлять параметры уже созданного заказа);
- DELETE - удаление данных на бэкенде (с помощью DELETE запроса можно удалять как данные (удалить ненужный заказ), так и файлы (фото, видео и прочее)).

Код HTTP-ответа - трехзначное число в строке состояния, которое показывает, успешно ли был выполнен запрос.
Коды бывают пяти классов:
1xx - информационные сообщения;
2xx - сообщения об успехе;
3xx - сообщения о перенаправлении;
4xx - сообщения о клиентской ошибке;
5xx - сообщения о серверной ошибке.

14. Опиши, из каких компонентов может состоять URL. За что отвечает каждый из них?

URL состоит из:
- схемы - протокола, по которому осуществляется передача данных (например HTTP или HTTPS);
- логина и пароля - они сообщают серверу, какой пользователь к нему обращается;
- символ @ - разделитель между логином:паролем и именем хоста:портом;
- имя хоста и порт - это доменное имя или же IP-адрес сервера, к которому обращается клиент;
- путь - расположение ресурса (нужен для боле точного и быстрого доступа к ресурсу);
- параметры запроса - нужны для передачи дополнительной информации на сервер, чтоб получить необходимые данные;
- якорь - дополнительный указатель, с помощбю которого можно сразу попасть в нужную часть веб-страницы.
Схема и имя хоста являются обязательными элементами URL.

15. Какие виды мобильных приложений бывают? В чём особенность каждого?

Виды мобильных приложений:
- Веб-приложение - приложение запускается в браузере устройства, для его запуска не требуется установка дополнительного ПО, для запуска требуется знать URL и активное интернет-соединение;
- Нативное приложение - приложение требует его установки на устройство, оно более оптимизировано по устройсво, лучше использует его ресурсы, также может использовать дополнительные инструменты устройства - камеру, различные датчики, GPS и тд, для доступа к некоторым функиям такого приложения интернет-соединения может не потребоваться;
- Гибридные приложения - представляет собой гибрид веб и нативного приложения - в нативной "оболочке" запускается веб-содержимое, как и веб-приложение гибрид требует соединения с интернетом, как и нативное приложение гибрид требует установки на устройство.

16. Чем эмулятор отличается от реального устройства? Кратко опиши недостатки и преимущества при тестировании.

Эмулятор представляет собой виртуальную версию мобильного устройства, запущенного на ПК с помощью ПО (эмулятора). Вследствие этого некотрые функции возможно эмулировать на ПК, некоторые - невозможно.

Преимущества:
- возможность начать тестирование даже не имея реального устройства на руках;
- возможность автоматизировать тестирование;
- широкий выбор эмулируемых устройств.

Недостатки:
- невозможность эмулирования некоторых инструментов устройства - гироскоп, датчик освещености, камера, NFC и др;
- производительность эмулятора ограничена производительностью ПК, на котором производится эмулирование (для наилучшего результата ПК дожен быть достаточно производительным).

17. Проверь, есть ли ошибки в JSON
Если да, напиши номер строки с ошибкой, опиши ошибку и предложи исправление.

```
01  { "Меню": {
02      "id": "1",
03      "value": Файл,
04      "list":
05          "items": {
06              "new_doc": {
07                  "value": "Новый",
08                  "onclick": "create_new_doc"},
09              "open_doc": {
10                  "value": "Открыть...",
11                  "onclick": "open_doc"},
12              "save_doc": {
13                  "value": "Сохранить",
14                  "onclick": "save_doc",
15              "save_as_doc"
16                  "Сохранить как...",
17                  "onclick": "save_as_doc"},
18              "print_option": {
19                  "value": "Параметры печати",
20                  "onclick":
21                      "show_print_option":{
22                          "Цвет":"Насыщенный",
23                          "Черно-белая печать?":"",
24                          "Размер печати":"А4"}}
25              }
26          }
27      }
28  }
```

Ссылка на иллюстрацию:

https://code.s3.yandex.net/qa/schemes/diploma-29.png

строка 3 - значение Файл должно быть в кавычках: "value" : "Файл",
строка 4 - пропущена фигурная скобка для значения ключа "list", должно быть "list" : {
строка 14 - не закрыты фигурные скобки для объекта "save_doc", должно быть "onclick" : "save_doc"},
строка 15 - пропущено двоеточие и фугурная скобка для объекта "save_as_doc", должно быть "save_as_doc" : {
строка 16 - пропущен ключ для значения "Сохранить как ...", должно быть "value" : "Сохранить как ...",
строка 20 - значение ключа "onclick" не взято в фигурные скобки, должно быть "onclick" : {
строка 24 - пропущена фигурная скобка для значение ключа "onclick", должно быть "Размер печати" : "А4" }}}

18. Что такое реляционная база данных? Чем она отличается от нереляционной?

Реляционная база данных - набор данных, который состоит из таблиц и связей между ними. Таблица - совокупность строк и столбцов, где столбцы - поля (характеристики и свойства сущности), а строки - записи (значения характеристик для данной сущности).
В нереляционной базе данных, в отличии от реляционной, не используются таблицы. В таких БД модель хранения завязана на типе хранимых данных, эти данные могут храниться в виде документов JSON, пар "ключ - значение" или в виде граф.

19. Напиши, какие виды JOIN бывают. В чем особенность каждого?

Виды JOIN - соединений:
- INNER JOIN - в ответе возвращает строки только на пересечении двух таблиц;
- LEFT OUTER JOIN - в ответе возвращает все строки левой таблицы, при выполнении строками из левой таблицы условия соединения они дополняются данными из правой таблицы;
- RIGHT OUTER JOIN - в ответе возвращает все строки правой таблицы, при выполнении строками из правой таблицы условия соединения они дополняются данными из левой таблицы;
- FULL OUTER JOIN - в ответе возвращает строки левой и правой таблицы, при выполнении строками правой и левой таблицей условия соединения они объединяются в одну строку.

Исходные данные для заданий ниже

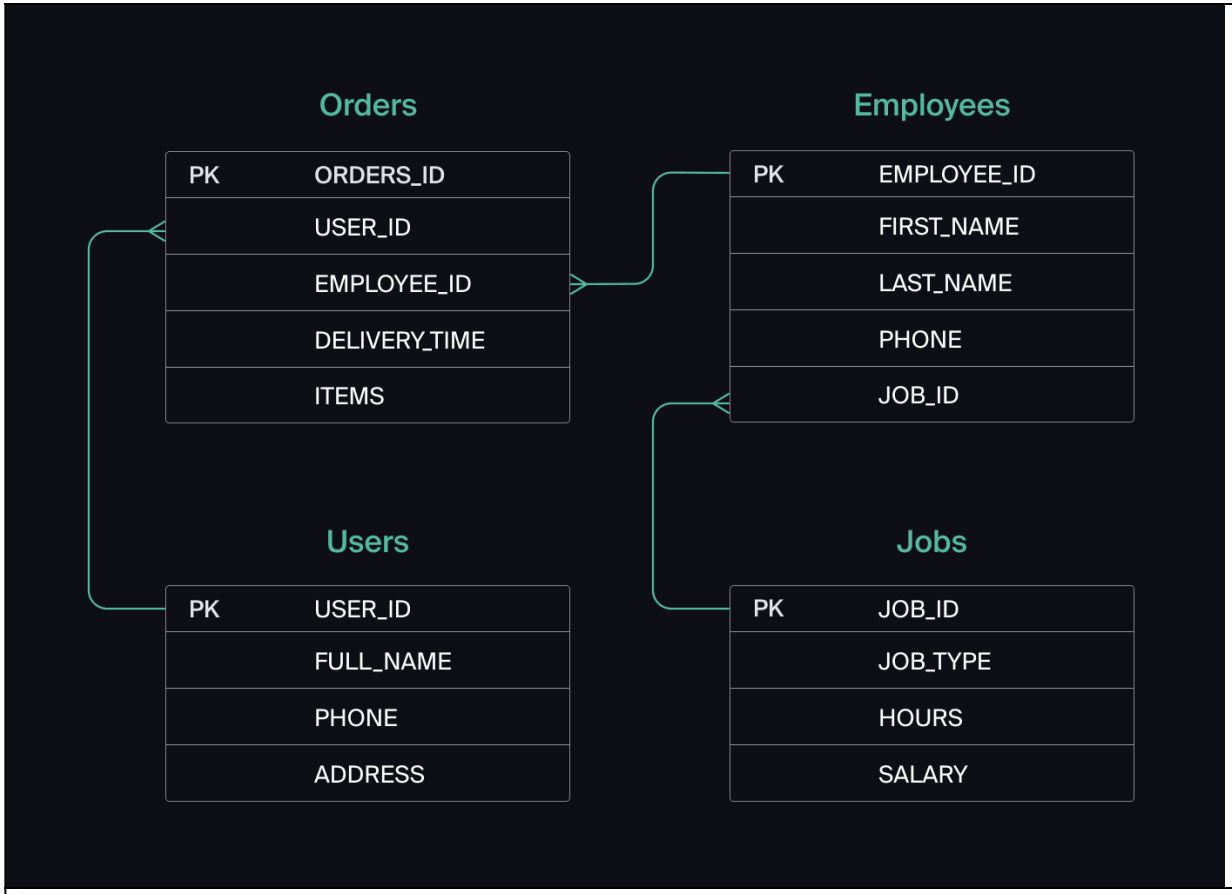
Ты тестируешь сервис, который доставляет еду за 30 минут. Пока это маленький стартап, поэтому ты работаешь всего с четырьмя таблицами:

Orders — все доставленные заказы;
ORDERS_ID — ID заказов, int;
USER_ID — ID пользователей, int;
EMPLOYEE_ID — ID сотрудников, int;
DELIVERY_TIME — время доставки в минутах, int;
ITEMS — список товаров, char;

Users — пользователи;
USER_ID — ID пользователей, int;
FULL_NAME — полное ФИО пользователя, char;
PHONE — номер телефона пользователя, char;
ADDRESS — адрес пользователя, char;

Employees — работники;
EMPLOYEE_ID — ID сотрудника, int;
FIRST_NAME —имя сотрудника, char;
LAST_NAME — фамилия сотрудника, char;
PHONE — телефон сотрудника, char;
JOB_ID — ID специализации, int;

Jobs — типы работ в сервисе
JOB_ID — ID специализации, int;
JOB_TYPE — тип специализации, char;
HOURS — число рабочих часов в неделю, int;
SALARY — зарплата сотрудника с данной специализацией в рублях, int;



Ссылка на иллюстрацию:

https://code.s3.yandex.net/qa/schemes/diploma-33.png

20. В службу поддержки пришло много жалоб: заказы, в которых есть гречка, доставляют почти час, хотя сервис обещает успеть в 30 минут.

Проверь, действительно ли курьеры опаздывают. Выбери все заказы, где есть хотя бы один товар - «гречка» и время доставки свыше 30 минут. В результирующей таблице должны быть ID заказов и ID курьеров.

В ответе приложи SQL-запрос.

```
SELECT
  ORDERS_ID AS ORDERS_ID,
  EMPLOYEE_ID AS EMPLOYEE_ID
FROM
  Orders
WHERE
  DELIVERY_TIME > 30 AND ITEMS LIKE '%гречка%';
```

21. Менеджер предложил добавить новую функциональность в продукт: мониторинг, который показывает самых активных клиентов за всё время работы компании.

Проверь, что список пользователей корректно выводится на экран. На этом этапе разработки достаточно проверить только ID клиентов.

Выбери пять самых активных клиентов по количеству заказов.

В результирующую таблицу выведи ID каждого пользователя и число заказов.

Отсортируй данные по убыванию числа заказов, выбери пять самых активных клиентов.

В ответе приложи SQL-запрос.

```
SELECT
  USER_ID AS USER_ID;
  COUNT(ORDERS_ID) AS cnt
FROM
  Orders
GRROUP BY
  USER_ID
ORDER BY
  cnt DESC
LIMIT
  5;
```

22. Из бухгалтерии пришёл баг-репорт: зарплаты сотрудников рассчитываются некорректно. Оказалось, что почти все ошибки в расчётах — в расчётных листах менеджеров.

Выведи список ID всех сотрудников, у которых в специализации содержится «менеджер», с зарплатой больше 70 000 рублей.

В ответе приложи SQL-запрос.

```
SELECT
  EMPLOYEE_ID AS EMPLOYEE_ID
FROM
  Employees
INNER JOIN Jobs ON Employee.JOB_ID = Jobs.JOB_ID
WHERE
  Jobs.JOB_TYPE LIKE '%менеджер%' AND Jobs.SALARY > 70000;
```

23. Изучи три ситуации и ответь на вопрос: стоит ли писать автотесты в этом случае? Аргументируй свой ответ.

1) Проект существует давно, у него написано много ручных тестов.

2) Проект временный: продлится всего несколько месяцев.

3) Проект нестабилен: в функциональность часто вносят изменения.

1 - Для давно существующего проекта стоит применять автотетсы, т.к. требования должны быть уже устоявшимися, известны проблемные места с багами для их более легкого обнаружения, в таких проектах часто проводят регрессию, здесь автотесты могут упростить работу тестировщикам.
2 - Для временного проекта автотесты применять не стоит, т.к. времени на полноценную разработку и отладку автотестов может и не хватить, логичнее будет обойтись ручным тестированием.
3 - При нестабильном прокте автотесты лучше не применять, т.к. при частых изменениях фнкционала надо будет вносить соответствующие правки в автотесты, что приведет к увеличению времени для разработки и отладки тестов и увеличению финансового расхода.