# Письменные ответы на вопросы

**Вопрос 1:**

Для составления тест-кейсов к задаче можно использовать попарное тестирование. Если составлять таблицу принятия решений, то количество проверок составит 360, и некоторые из них будут дублировать друг друга по содержанию и будут излишними. Попарное тестирование позволит сократить количество проверок до 61, при это все параметры буду проверены:

|  | **ЖК** | **Тип квартиры** | **Этаж** | **Отделка** | **Балкон** |
| --- | --- | --- | --- | --- | --- |
| **1** | Золотое сияние | Студия | 1 | Чистовая | Есть |
| **2** | Золотое сияние | 4-комн | 5 | Чистовая | Есть |
| **3** | Небесный полет | 4-комн | 6 | Чистовая | Нет |
| **4** | Золотое сияние | 4-комн | 7 | Чистовая | Нет |
| **5** | Золотое сияние | Студия | 8 | Чистовая | Есть |
| **6** | Золотое сияние | 1-комн | 9 | Черновая | Нет |
| **7** | Золотое сияние | 2-комн | 1 | Чистовая | Есть |
| **8** | Золотое сияние | 3-комн | 2 | Черновая | Нет |
| **9** | Золотое сияние | 1-комн | 4 | Черновая | Нет |
| **10** | Небесный полет | 3-комн | 7 | Чистовая | Нет |
| **11** | Небесный полет | 4-комн | 8 | Черновая | Есть |
| **12** | Небесный полет | Студия | 9 | Чистовая | Нет |
| **13** | Небесный полет | 1-комн | 1 | Черновая | Есть |
| **14** | Небесный полет | 2-комн | 2 | Чистовая | Нет |
| **15** | Небесный полет | 3-комн | 3 | Черновая | Есть |
| **16** | Небесный полет | Студия | 4 | Чистовая | Есть |
| **17** | Небесный полет | 1-комн | 5 | Черновая | Нет |
| **18** | Небесный полет | 2-комн | 6 | Чистовая | Есть |
| **19** | Золотое сияние | 4-комн | 9 | Черновая | Нет |
| **20** | Золотое сияние | 1-комн | 2 | Черновая | Есть |
| **21** | Золотое сияние | 2-комн | 3 | Чистовая | Нет |
| **22** | Золотое сияние | 3-комн | 4 | Черновая | Есть |
| **23** | Золотое сияние | Студия | 5 | Чистовая | Нет |
| **24** | Золотое сияние | 1-комн | 6 | Чистовая | Есть |
| **25** | Золотое сияние | 2-комн | 7 | Черновая | Нет |
| **26** | Золотое сияние | 3-комн | 8 | Чистовая | Есть |
| **27** | Небесный полет | Студия | 2 | Чистовая | Есть |
| **28** | Небесный полет | 1-комн | 3 | Черновая | Нет |
| **29** | Небесный полет | 2-комн | 4 | Чистовая | Есть |
| **30** | Небесный полет | 3-комн | 5 | Чистовая | Нет |
| **31** | Небесный полет | Студия | 6 | Черновая | Есть |
| **32** | Небесный полет | 1-комн | 7 | Чистовая | Нет |
| **33** | Небесный полет | 2-комн | 8 | Черновая | Есть |
| **34** | Небесный полет | 3-комн | 9 | Чистовая | Есть |
| **35** | Небесный полет | 4-комн | 1 | Черновая | Нет |
| **36** | Золотое сияние | 2-комн | 5 | Черновая | Нет |
| **37** | Золотое сияние | 3-комн | 6 | Чистовая | Есть |
| **38** | Золотое сияние | Студия | 7 | Черновая | Есть |
| **39** | Золотое сияние | 1-комн | 8 | Чистовая | Нет |
| **40** | Золотое сияние | 2-комн | 9 | Черновая | Есть |
| **41** | Золотое сияние | 3-комн | 1 | Чистовая | Нет |
| **42** | Золотое сияние | 4-комн | 2 | Черновая | Есть |
| **43** | Золотое сияние | Студия | 3 | Чистовая | Нет |
| **44** | Небесный полет | 2-комн | 6 | Черновая | Нет |
| **45** | Небесный полет | 3-комн | 7 | Чистовая | Есть |
| **46** | Небесный полет | Студия | 8 | Черновая | Нет |
| **47** | Небесный полет | 1-комн | 9 | Чистовая | Есть |
| **48** | Небесный полет | 2-комн | 1 | Черновая | Нет |
| **49** | Небесный полет | 3-комн | 2 | Чистовая | Есть |
| **50** | Небесный полет | 4-комн | 3 | Чистовая | Нет |
| **51** | Небесный полет | Студия | 4 | Черновая | Есть |
| **52** | Небесный полет | 1-комн | 5 | Чистовая | Есть |
| **53** | Золотое сияние | 3-комн | 8 | Чистовая | Нет |
| **54** | Золотое сияние | Студия | 9 | Черновая | Есть |
| **55** | Золотое сияние | 1-комн | 1 | Чистовая | Нет |
| **56** | Золотое сияние | 2-комн | 2 | Чистовая | Есть |
| **57** | Золотое сияние | 3-комн | 3 | Черновая | Есть |
| **58** | Золотое сияние | 4-комн | 4 | Чистовая | Нет |
| **59** | Золотое сияние | Студия | 5 | Черновая | Есть |
| **60** | Золотое сияние | 1-комн | 6 | Чистовая | Нет |
| **61** | Золотое сияние | 2-комн | 7 | Черновая | Есть |
|  |  |  |  |  |  |

**Вопрос 2:**

1. Тест-кейс не содержит уникальный номер (ID), следовательно, на него невозможно будет ссылаться при проверке.
2. Заголовок некорректно описывает, что именно проверяется: главная страница Яндекс Практикума должна что-то содержать или не содержать для неавторизованного пользователя? Какие особенности главной страницы мы проверяем данной проверкой? Главная страница не должна быть доступна неавторизованному пользователю?
3. Не указано окружение, в котором должно проводиться тестирование.
4. Шаги воспроизведения - я бы перефразировала шаги воспроизведения, чтобы они выглядели как конкретные действия.
5. Ожидаемый результат - так как из заголовка не совсем понятно, что именно мы хотели проверить, то и ожидаемый результат сформулирован обобщенно. Можно приложить скриншот экрана, чтобы было понятно, какой результат проверки является ожидаемым, либо ссылку на макет в Figma.
6. Отсутствует статус проверки. Необязательный атрибут, может отсутствовать, если хранение тест-кейса не позволяет его оставить.
7. Не указан приоритет теста.
8. Лучше оставить столбец для ссылки на баг-репорт и комментариев.

Я бы скорректировала тест-кейс следующим образом:

**ID:** ЯП1

**Заголовок:** Главная страница Яндекс Практикума доступна неавторизованному пользователю при переходе по ссылке на сайт Яндекс Практикума

**Предусловие:** Пользователь не авторизован в системе Яндекс Практикум

**Шаги воспроизведения:**

1. Октрыть сайт Яндекс Практикум по ссылке <https://practicum.yandex.ru/>

**Ожидаемый результат:** Открывается главная страница Яндекс Практикум в соответствии с приложенным макетом в Figma.

**Окружение:** Яндекс.Браузер, разрешение 800х600

**Статус (необязательный атрибут):** Passed/Failed/Skipped (выбрать)

**Ссылка на баг-репорт:** (если есть баг)

**Комментарии:** ссылка на макет в Figma или его скрин.

**Вопрос 3:**

Для того, чтобы понять, где возникает ошибка, необходимо проследить как передаются данные от фронтенда, бэкенда до базы данных, то есть как обрабатываются данные на каждом из этих этапов, и определить, на каком этапе возникает ошибка.

Для проверки корректности обработки запроса фронтендом в DevTools открываем вкладку Элементы (Elements). Здесь необходимо найти HTML-запрос формы на создание заказа и проверить, что чек-бокс “Заплатить сразу” отмечен (проставлен атрибут checked). Если атрибут не отмечен/отсутствует или равен false, то проблема на стороне фронтенда.

На вкладке Сеть (Network) находятся запросы фронтенда и ответы бэкенда. Нужно найти запрос создания заказа и проверить эту запись: POST-запрос должен содержать все необходимые заголовки, параметры и тело запроса (в соответствии с технической документацией), в том числе проверить, как передается в запросе чек-бокс “Заплатить сразу” (указано true или false), и посмотреть код ответа сервера. Если чек-бокс “Заплатить сразу” в запросе true, а код ответа сервера 200 или 201, то нужно искать ошибку в базе данных.

**Вопрос 4:**

Метод GET используется для запроса информации с сервера, без изменения состояния сервера. Он не предназначен для изменения или удаления данных на сервере. Следовательно, ситуация, когда отправленный GET-запрос удаляет всю информацию в базе данных, является маловероятной, и может быть результатом умышленных действий злоумышленников.

Разработчик может удалить данные в БД через GET-запрос. Для этого необходимо записать в код операцию удаления данных, содержащих параметр, который передается в GET-запросе. В этом случае, когда клиент передаст GET-запрос на сервер, сервер запросит их в БД, и удалит эти данные в момент их получения.

Также удаление может быть произведено в совокупности случаев, когда имеет место некорректная работа сервера, не соблюдаются стандарты HTTP-протокола, имеются ошибки в программном коде сервера или сервер скомпрометирован злоумышленниками, найдены уязвимости, которые позволяют выполнять изменение и удаление данных при обработке GET-запроса.

**Вопрос 5:**

Первичный ключ - это уникальный признак каждой записи в базе данных, позволяющий избежать создания одинаковых записей.

Для таблицы employee первичным ключом является поле id, для таблицы position - поле id. Id в обоих случаях является уникальным идентификатором, которые позволяет записям в таблицах не быть одинаковыми, по id можно идентифицировать любого сотрудника и любую должность.

Fio, name и salary не могут быть первичными ключами, так как могут повторяться у разных значений таблицы.

Внешний ключ - это поле, которое отсылает к первичному ключу другой таблицы. Таким ключом может выступать поле position\_id таблицы employee, потому что это поле является связующим с полем id в таблице position.

**Вопрос 6:**

SELECT e.fio,

p.name,

p.salary

FROM employee AS e

JOIN position AS p ON e.position\_id = p.id;

**Вопрос 7:**

Объявлена функция test\_integer\_division, которая рассчитывает значение переменной “а” путем целочисленного деления (используется оператор “//”) 5 на 2. Если деление выполняется верно, то переменная “a” должна быть равна 2, то есть мы должны проверить выполнение логического оператора равенства “==”, что значение переменной “а” равно числу 2. Для того, чтобы увидеть, выполняется ли данное равенство, мы можем использовать функцию print().

Мы можем вывести значение равенства, то есть получить значение логического выражения (True):

def test\_integer\_division():

a = 5//2;

print(a==2)

test\_integer\_division();

либо мы можем вывести значение переменной “а” и увидеть, что оно равно 2:

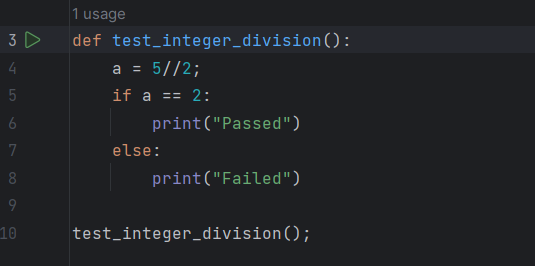
def test\_integer\_division():

a = 5//2;

print(a)

test\_integer\_division();

По префиксу “test” Python понимает, что это тест. Чтобы метод test\_integer\_division() стал полноценным тестом и что-то проверял, можно добавить условный оператор if, который будет проверять значение переменной “а” (равно ли оно “2”). Тогда, если (if) “а” равно 2, то пусть результат будет “Passed”, а если не равно (else), то пусть результат будет “Failed”:



Таким образом, решить данную задачу можно несколькими способами. В любом из них лучше дополнить функцию print(), чтобы результат был виден на экране.

**Вопрос 8:**

Классы эквивалентности и граничные значения могут существовать по отдельности, если классы эквивалентности представлены не диапазоном, а набором.

Граничные значения - это первая и последняя точки класса эквивалентности. Но если класс - это набор, то в класс относят все значения, входящие в него. У таких классов границ не бывает, как например у списка праздников, списка видов животных при выборе корма (для petshop), а также перечня производителей корма (для влажного и сухого корма), списка уроков (для электронного дневника) или списка классов (все первые классы, но значения 1А, 1Б, 1В и т.д.), списки пользователей (зарегистрированные и незарегистрированные).

**Вопрос 9:**

При тестировании значения на границах, входящих в диапазон, мы проверяем, что результат проверки соответствует заявленному в требованиях для этого диапазона. Наиболее частые ошибки встречаются как раз на границах диапазона (например, если забыть включить это значение в диапазон, пропустив знак равенства, или наоборот, исключить его, поставив знак неравенства или просто равно). Можно исключить проверку в середине диапазона в пользу проверок на 1 шаг внутрь и 1 шаг вне диапазона, но также необходимо проверить граничное значение диапазона. При проверке значений около граничного значения диапазона мы проверяем, что в диапазон попадают не только значения на границах диапазона, но и значения внутри диапазона (т.е. мы получаем одинаковый результат, соответствующий заявленным в требованиях), и не попадают значения за пределами диапазона (т.е. мы получаем результат, отличный от результата, заявленного в требованиях для этого диапазона).

Если мы провели проверки граничного значения, значения на 1 шаг внутрь диапазона и на 1 шаг за пределами диапазона, то мы проверили этот диапазон, и тестирование значения в середине диапазона будет излишним, так как это значение принадлежит к одному классу и будет дублировать проверку значения на 1 шаг внутрь диапазона.

**Вопрос 10:**

Если разработчик просит воспроизвести баг и прислать логи, то необходимо сделать следующее:

Вариант 1, с помощью Logcat в Android Studio:

1. Переводим смартфон в режим разработчика.
2. Подключаем устройство по USB к компьютеру.
3. Заходим на главную страницу Android Studio.
4. На консоли кликаем Logcat.
5. Из списка выбираем смартфон, на котором мы тестировали приложение Яндекс Аренда.
6. Выбираем приложение Яндекс Аренда.
7. Выделяем текст лога, копируем его и вставляем в текстовый документ, сохраняем в файл.

Вариант 2, с помощью ADB:

1. Переводим смартфон в режим разработчика.
2. Выбираем “Отладку по USB”.
3. Подключаем устройство по USB к компьютеру.
4. Заходим в [Android Debug Bridge](https://developer.android.com/studio/command-line/adb).
5. Открываем терминал.
6. adb devices - проверяем подключенное устройство.
7. adb -s название устройства logcat - команда выводит все логи в консоль в режиме реального времени.
8. db logcat > c:\adb\log.txt - указываем путь к файлу, в который будут записываться логи.
9. Воспроизводим на устройстве последовательность шагов:

9.1. Открываем приложение Яндекс Аренда.

9.2. Находим вариант для аренды.

9.3. Переходим в раздел “Показать на карте”.

Ожидаемый результат: объект аренды отображается на карте.

Фактический результат: появляется сообщение «Непредвиденная ошибка», и приложение закрывается.

1. Прикладываем файл log.txt, в который сохранились логи от данной ошибки.
2. Чтобы остановить запись логов в файл нажимаем сочетание клавиш ctrl+c.

В качестве воспроизведения бага записываем баг-репорт и прикладывает файл с логами:

**Заголовок:** Непредвиденная ошибка закрывает приложение Яндекс Аренда при переходе в раздел “Показать на карте”

**Описание:** При переходе в раздел “Показать на карте” в приложении Яндекс Аренда возникает сообщение “Непредвиденная ошибка”, и приложение закрывается.

**Шаги воспроизведения:**

1. Открываем приложение Яндекс Аренда.

2. Находим вариант для аренды.

3. Переходим в раздел “Показать на карте”.

**Ожидаемый результат:** объект аренды отображается на карте.

**Фактический результат:** появляется сообщение «Непредвиденная ошибка», и приложение закрывается.

**Окружение**: ОС Windows 10 Pro версия 22н2, Firefox Browser, версия 111.0.1 (64-разрядный), разрешение 1920х1080.

**Вложения:** Файл с логами.

Отчет о тестировании

## Функциональное тестирование веб-приложения

Приложение проверено на стенде <https://ab86d2e7-a110-4255-bd61-f11148e5826e.serverhub.praktikum-services.ru/>

Все известные требования были покрыты чек-листом.

Результаты выполнения тестов можно посмотреть в приложенной таблице Excel Ольга Каплина, 9-я когорта – дипломная работа.

Из 526 тестов успешно прошло 415, не прошло — 111.

Список багов, найденных при тестировании, разбит по приоритетам:

1. Блокирующие:

отсутствуют.

2. Критичные:

<https://olkatester.youtrack.cloud/issue/YS-14>

<https://olkatester.youtrack.cloud/issue/YS-12>

3. Серьезный приоритет:

https://olkatester.youtrack.cloud/issue/YS-6

<https://olkatester.youtrack.cloud/issue/YS-13>

<https://olkatester.youtrack.cloud/issue/YS-24>

4. Обычный приоритет:

<https://olkatester.youtrack.cloud/issue/YS-3>

<https://olkatester.youtrack.cloud/issue/YS-5>

<https://olkatester.youtrack.cloud/issue/YS-7>

<https://olkatester.youtrack.cloud/issue/YS-8>

<https://olkatester.youtrack.cloud/issue/YS-9>

<https://olkatester.youtrack.cloud/issue/YS-11>

<https://olkatester.youtrack.cloud/issue/YS-15>

<https://olkatester.youtrack.cloud/issue/YS-16>

<https://olkatester.youtrack.cloud/issue/YS-17>

<https://olkatester.youtrack.cloud/issue/YS-18>

<https://olkatester.youtrack.cloud/issue/YS-19>

<https://olkatester.youtrack.cloud/issue/YS-20>

<https://olkatester.youtrack.cloud/issue/YS-21>

<https://olkatester.youtrack.cloud/issue/YS-22>

<https://olkatester.youtrack.cloud/issue/YS-25>

<https://olkatester.youtrack.cloud/issue/YS-26>

5. Низкий приоритет:

<https://olkatester.youtrack.cloud/issue/YS-1>

[https://olkatester.youtrack.cloud/issue/YS-2](https://olkatester.youtrack.cloud/issue/YS-2/Razmer-knopki-Status-zakaza-v-hedere-stranicy-Dlya-kogo-samokat-ne-sootvetstvuet-trebovaniyam)

h[ttps://olkatester.youtrack.cloud/issue/YS-10](https://olkatester.youtrack.cloud/issue/YS-10/Cvet-plejsholdera-polya-Stanciya-metro-otlichaetsya-ot-cveta-plejsholderov-drugih-polej-stranicy)

<https://olkatester.youtrack.cloud/issue/YS-23>

Заключение:

1. Самым критичным является баг в браузере Google Chrome, который не позволяет оформить заказ при корректном заполнении всех полей форм. Это потеря даже самых лояльных клиентов, которые готовы “закрыть глаза” на другие недоработки сайта.
2. Самая «хитрая» серая зона - каким образом пользователь должен узнать сколько стоит его заказ?
3. Яндекс Самокат в данном виде не готов к релизу. Необходимо исправить следующие баги: возможность создания заказа в Google Chrome при корректном заполнении всех обязательных полей, ввод только корректной даты заказа (сейчас есть возможность оформить заказ на прошедшие даты), ввод только корректного номера телефона (сейчас есть возможность вводить номер +0000000000). Также необходимо исправить баги самого заказа - на странице заказа отображается неверная дата заказа (не та, которая была выбрана на этапе создания заказа). Задание не включает тестирование карточки заказа, но был замечен такой баг.

## Ретест багов в мобильном приложении

Был проверен фикс багов. Из них не исправлено 1, исправлено — 3, выявлено новых багов - 1.

Список багов можно посмотреть здесь:

<https://olkatester.youtrack.cloud/issue/MYS-1>

<https://olkatester.youtrack.cloud/issue/MYS-2>

<https://olkatester.youtrack.cloud/issue/MYS-3>

<https://olkatester.youtrack.cloud/issue/MYS-4>

https://olkatester.youtrack.cloud/issue/MYS-5

Регрессионное тестирование мобильного приложения по готовым тест-кейсам

Результаты выполнения регрессионных тестов можно посмотреть в таблице Excel Ольга Каплина, 9-я когорта – регрессионное тестирование Самокат.

Список багов, найденных при тестировании, разбит по приоритетам:

1. Блокирующие:  
отсутствуют.

2. Критичные:  
<https://olkatester.youtrack.cloud/issue/MYS-5>

<https://olkatester.youtrack.cloud/issue/MYS-6>

<https://olkatester.youtrack.cloud/issue/MYS-7>

3. Средний приоритет:  
<https://olkatester.youtrack.cloud/issue/MYS-8>

<https://olkatester.youtrack.cloud/issue/MYS-9>

<https://olkatester.youtrack.cloud/issue/MYS-10>

https://olkatester.youtrack.cloud/issue/MYS-11

4. Низкий приоритет:  
отсутствуют.

Заключение:

1. Самым критичным показался баг с задвоением заказов - курьеры привезут в 2 раза больше самокатов клиентам.
2. Продукт может быть выпущен в релиз только после доработок. Необходимо устранить задвоение заказов и скорректировать полные карточки заказов таком образом, чтобы курьер мог видеть всю информацию, переданную клиентом. Кроме того, приложение не стабильно - если не использовать в течение 10 минут, то оно вылетает (на экране появляется страница “поломки” приложения, исправление происходит при нажатии на кнопку “Перезапустить”).