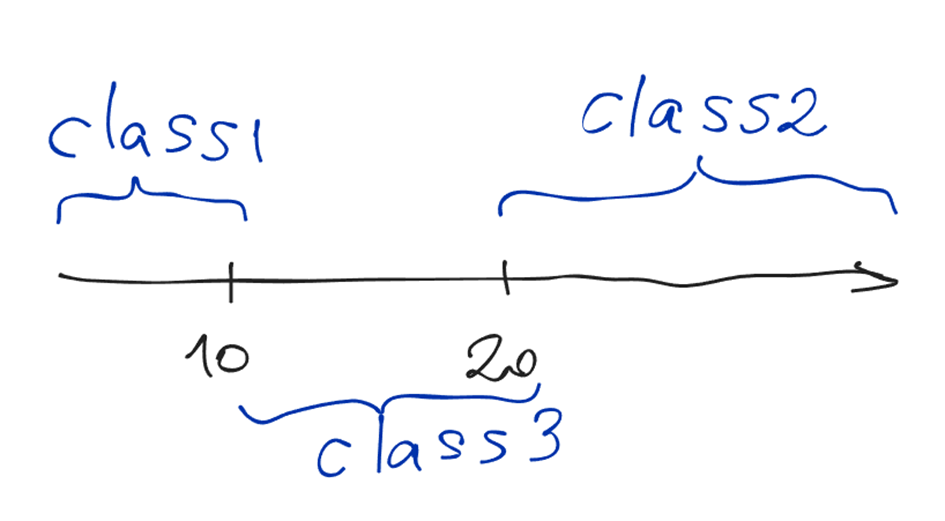
<https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/>

Шпаргалка по техникам тест дизайна

Добро пожаловать!

Ежедневной задачей инженера по контролю качества (QA Engineer) является создание тест-кейсов для проверки требований продукта. В этой статье я собрал для вас техники проектирования тестов, которые помогут оптимизировать ваш набор тестов.

* **#1 Классы эквивалентности**
* [Эквивалентное разделение (Equivalence Partitioning)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#1)



Является основной техникой проектирования тестов, которую должен использовать каждый инженер по контролю качества.

Смысл этого подхода заключается в выборе значений, представляющих различные классы тестовых данных, чтобы мы могли проверить требования к продукту.

**Пример**

У меня есть калькулятор инвестиций, который содержит поле процента прибыли, которое мне необходимо указать. Разрешенные значения находятся в диапазоне от 1 до 100 по требованию заказчика.

**Шаг 1:** Определите эквивалентные классы:

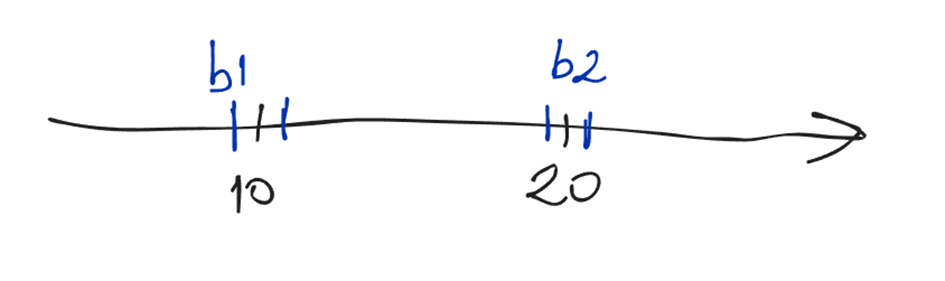
* Положительные числа (1–100)
* Отрицательные числа и ноль (<=0)
* Больше допустимого (>100)
* Буквы (a-z, a-Z, кириллица, арабский и т. д.)
* Специальные символы (!@# и т. д.)

**Шаг 2:** Выберите одно значение для каждого класса. И у нас есть следующие тест-кейсы для проверки: 50, -10, 146, aBc, !

**Где использовать?**

Везде! При тестировании пользовательского интерфейса (UI) - это поля, даты, конкретные кнопки. При тестировании API нам нужно проверить все возможные параметры в теле запроса (body), заголовках (headers), пути (path) или параметрах запроса (query parameters).

* **#2 Граничные значения**
* [Граничные значения (Boundary Values)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#2)



Эта техника является "братом" разбиения на классы эквивалентности.

Смысл этого подхода заключается в выборе значений на границах эквивалентных классов с минимальным шагом.

**Пример:**

Интернет-магазин предоставляет скидку 10%, если сумма покупки превышает $100.

**Шаг 1:** Определите эквивалентные классы и границы каждого из них:

* Сумма покупки <= $100
* Сумма покупки > $100

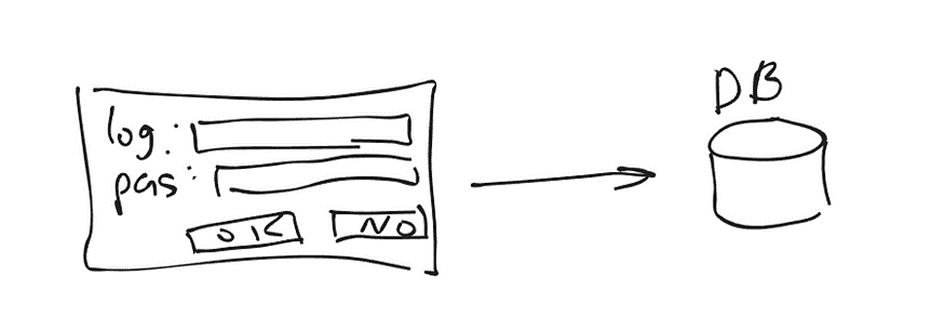
**Шаг 2:** Выберите значения, находящиеся на границе этих классов с минимальным отклонением, а также само граничное значение:

* Сумма покупки = $99.99: скидка не применяется
* Сумма покупки = $100: скидка не применяется
* Сумма покупки = $100.01: скидка применяется

**Где использовать?**

Ответ тот же, что и для разбиения на классы эквивалентности - везде.

**#3 Причинно-следственный анализ**



Каждое приложение ведет себя схоже — производит определённый отклик на каждое действие пользователя.

Этот подход заключается в систематизации карты влияния на приложение. Знание того, какое условие должно привести к какому-либо отклику, может помочь нам оптимизировать количество тест-кейсов и обеспечить соответствующее покрытие.

**Пример:**

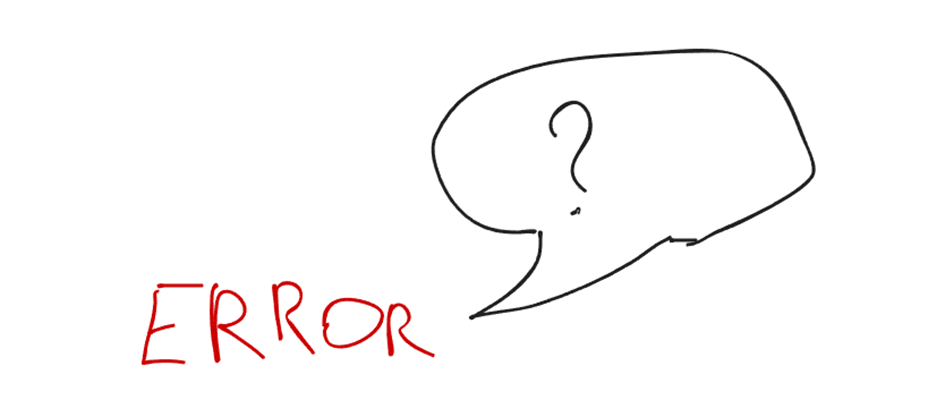
Как пользователь, я заполняю поля логина и пароля и нажимаю кнопку OK. Я ожидаю, что мои данные пользователя будут сохранены в базе данных. Это простой пример, чтобы помочь вам понять технику. Более сложные случаи могут быть рассмотрены в будущем.

* **Шаг 1**: Определите карту причин и откликов.
* **Шаг 2**: Создайте таблицу принятия решений на основе этой карты.

**Где использовать?**

Эта техника может быть использована в случаях, когда у нас есть неочевидные зависимости и сложные условия для принятия решения. Она также может быть применена, когда наши действия влияют на хранение данных или другие внешние сервисы. Стоит отметить, что эта техника хорошо сочетается с диаграммами состояний и последовательности.

**#4 Прогнозирование** **ошибок**



Этот подход основан на вашем предыдущем опыте использования других аналогичных приложений / платформ. Предполагается, что вы знаете некоторые ситуации, которые могут вызвать ошибки и запутать пользователя с неожиданными результатами.

**Пример**

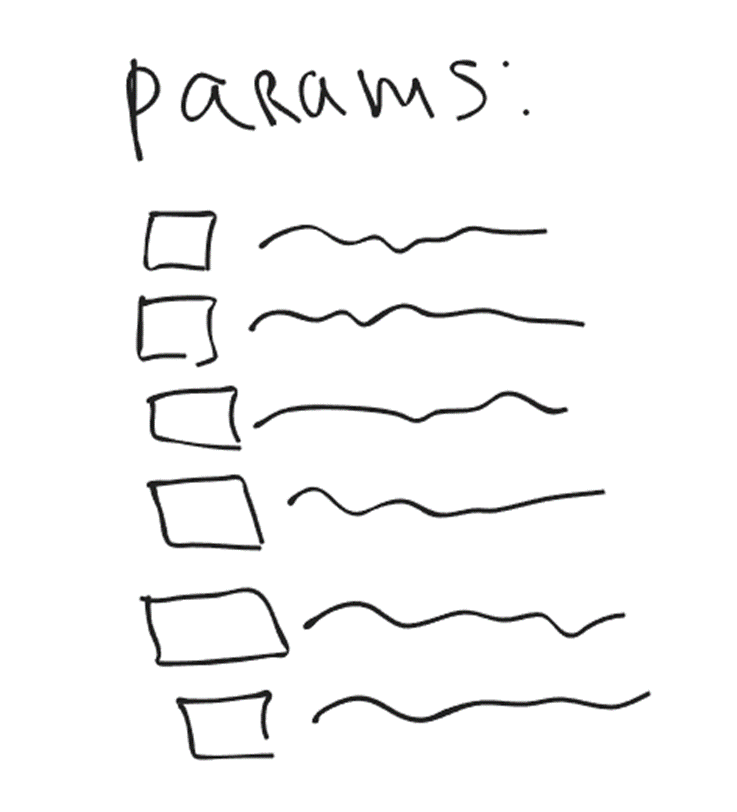
В предыдущем сценарии мы можем не предоставлять данные вообще, предоставлять специальные символы в качестве имени пользователя, только цифры и т. д.

* **Шаг 1**: Определите отрицательные случаи для функциональности, которую нужно протестировать на основе каждой техники проектирования тестов.
* **Шаг 2**: Определите, что ожидается для каждого отрицательного сценария из шага #1.

**Где использовать?**

Везде, где пользователь может предоставлять данные. Для UI это могут быть формы, флажки, кнопки и т. д. Для API-сервисов - параметры запроса, параметры пути, параметры тела.

**#5 Попарное тестирование**



Этот подход основан на большом количестве входных параметров. Чем больше параметров, тем больше вероятность ошибки. Наша цель как специалиста по тестированию — сократить количество тест-кейсов до оптимального. Подход парного тестирования поможет нам в этом.

Суть его заключается в том, чтобы рассмотреть все возможные комбинации каждой пары входных параметров.

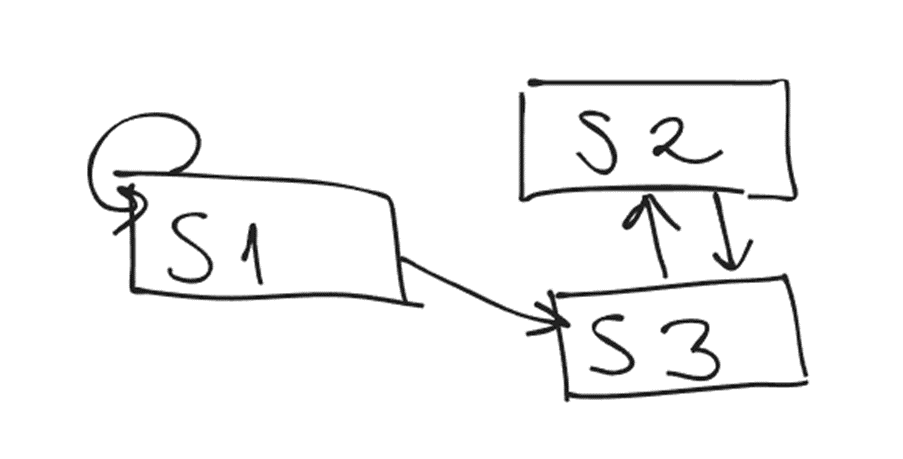
* **Шаг 1:** Определите возможные параметры и их значения для функциональности, которую необходимо протестировать.
* **Шаг 2:** Создайте тест-кейсы, чтобы включить все возможные пары между каждыми 2 параметрами. Вы можете использовать инструменты, такие как <https://pairwise.teremokgames.com/>, чтобы облегчить свою работу.

**Где использовать?**

Когда у вас большое количество входных параметров и большое количество возможных значений параметров. Это может быть применено к приложениям с графическим интерфейсом и API-приложениям.

Ее стоит **использовать в том случае**, когда входные данные связаны друг с другом. Точнее результат выполнения теста напрямую зависит от того, какие комбинации данных будут подаваться на входе.

**#6 Диаграмма состояний**



Тестирование приложения связано с последовательностью экранов (страниц), созданием/чтением/обновлением/удалением разных типов объектов. Диаграммы состояний могут помочь нам охватить все ветви для таких объектов и экранов.

Смысл данного подхода заключается в создании карты переходов для каждого типа объекта и создании набора тестов, охватывающих все переходы между состояниями.

**Пример:**

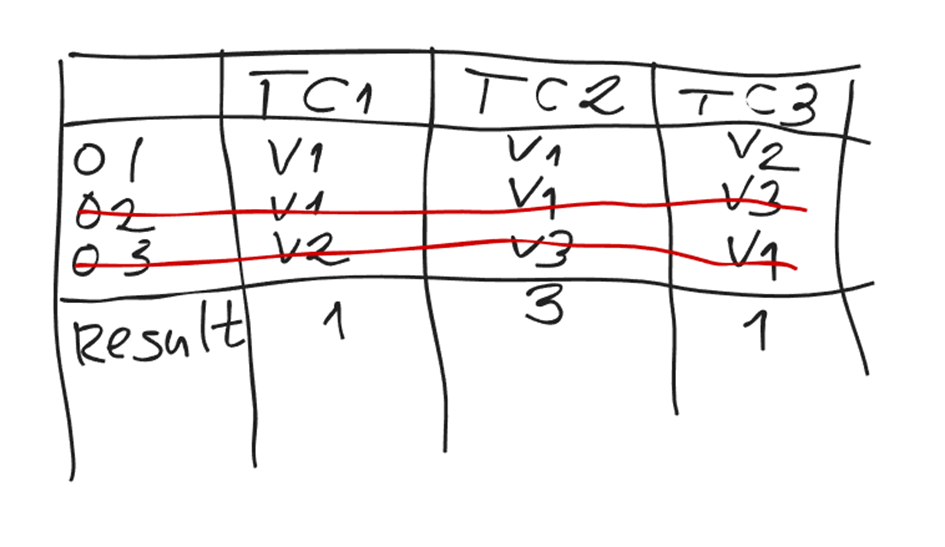
Пользователь имеет возможность написать комментарий к посту. Затем он может изменить или удалить этот комментарий. У комментариев есть несколько состояний: создан, изменен, удален. Каждое из этих состояний может быть характеризовано набором уникальных свойств.

* **Шаг 1**: Определите объект, который мы хотим протестировать.
* **Шаг 2:** Нарисуйте прямоугольник для каждого состояния, определенного для объекта.
* **Шаг 3:** Создайте переходы между состояниями и отметьте каждый из них связанным действием.
* **Шаг 4:** Вы готовы создавать положительные и отрицательные тест-кейсы.

**Где использовать?**

Прежде всего, при тестировании моделей объектов домена. Этот метод можно применять и к части пользовательского интерфейса, как уже упоминалось ранее. Мы можем охватить все переходы между экранами (страницами) пользовательского интерфейса и создать тестовые случаи, проверяющие переключение между ними.

* **#7 Таблица принятия решений**
* [Таблица принятия решений (Desicion Table)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#3)



Эта техника помогает наглядно изобразить комбинаторику условий из требований. Это помогает нам сократить количество ненужных тестов и предоставить наиболее эффективный набор тестов.

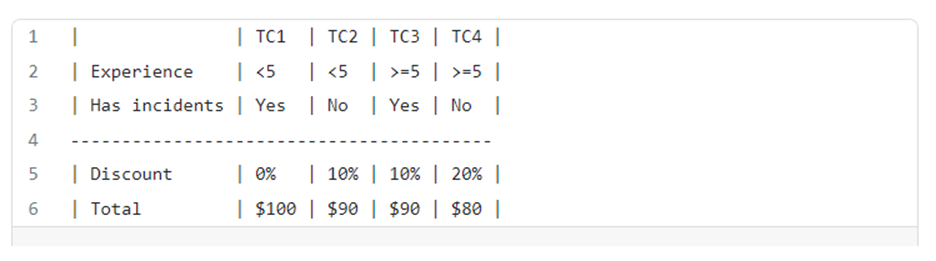
Этот подход помогает нам увидеть зависимости между тестовыми данными и конечным результатом и увидеть, действительно ли нужны ли нам некоторые сценарии, потому что они могут быть покрыты другими.

**Пример**

Водитель хочет купить страховку. Цена зависит от общего опыта и прошлых инцидентов. Допустим, если у меня есть 5 лет опыта, я могу получить скидку 10% от базовой цены (100 долларов), а если у меня не было аварий за последний год, то я могу получить еще 10%.

* **Шаг 1:** Подготовьте список входных параметров с возможными доступными значениями:
* **Шаг 2:** Создайте строки с каждым именем параметра в первом столбце.
* **Шаг 3:** Создайте тестовую таблицу на основе всех возможных комбинаций параметров и заполните эти данные в следующих столбцах таблицы.
* **Шаг 4:** Попробуйте найти тестовые примеры, которые «дублируются», например, где результаты зависят от 1 параметра, а остальные параметры не имеют значения. Эти тестовые случаи могут быть исключены из окончательного набора тестов.

См. такую таблицу ниже. Там мы не могли исключить ни одного теста, потому что пример очень простой и понятный, но в реальных задачах мы можем иметь 20 параметров со сложной логикой:



**Где использовать?**

Таблица решений может описывать сложные правила/требования. Условия — это входные данные, действия — это ожидаемый результат, а столбцы — тестовые примеры.

**Заключение**

В этой статье я создал для вас шпаргалку по техникам тест дизайна. Каждую из них, конечно, следует рассмотреть более подробно. В любом случае, эта шпаргалка поможет вам запомнить шаги для разработки набора тестов, если вы по каким-то причинам забудете их.

Вот список техник, которые я упомянул:

1. Классы эквивалентности
2. Граничные значения
3. Причинно-следственный анализ
4. Прогнозирование ошибок
5. Попарное тестирование
6. Диаграмма состояний
7. Таблица принятия решений

Желаю удачи и не останавливайтесь тестировать!

**Хабы:**

* [Тестирование IT-систем](https://habr.com/ru/hubs/it_testing/)
* [Тестирование веб-сервисов](https://habr.com/ru/hubs/web_testing/)
* [Тестирование мобильных приложений](https://habr.com/ru/hubs/mobile_testing/)
* [Тестирование игр](https://habr.com/ru/hubs/game_testing/)

**+8**

**111**

[**4**](https://habr.com/ru/articles/740026/comments/)

**[+4](https://habr.com/ru/articles/740026/comments/)**

**Комментарии 4**

[[](https://habr.com/ru/users/stopper79/)](https://habr.com/ru/users/stopper79/" \o "stopper79)

[**stopper79**](https://habr.com/ru/users/stopper79/)[7 июн в 00:07](https://habr.com/ru/articles/740026/#comment_25623246)

Я бы все таки, в

1. Граничные значения
2. Классы эквивалентности добавил проверку 0, иногда такие неожиданные результаты :)

**+3**

Ответить

[[https://habrastorage.org/r/w48/getpro/habr/avatars/9cd/e54/d3d/9cde54d3de1b4981d000e6a2f98b775d.png](https://habr.com/ru/users/KeyMan/)](https://habr.com/ru/users/KeyMan/" \o "KeyMan)

[**KeyMan**](https://habr.com/ru/users/KeyMan/)[26 июн в 10:34](https://habr.com/ru/articles/740026/#comment_25688784)

Согласен с данным комментарием, что ноль очень важен. Но определение классов эквивалентов это отдельная тема, в принципе, автор данной статьи в заключение это и сказал.

А так для целочисленных значений, в диапазоне от 10 до 20, классами эквивалентности будут:

А для диапазона от -10 до 10, классами эквивалентности будут:

Для строковых типов данных все еще интереснее) Есть различные символы, включая и спецсимволы, есть кодировки (например, ASCII и UTF-8), также есть XSS инъекции и прочее. В данном случае начинает работать принцип "тестирование зависит от контекста", классы эквивалентности будет завесить от того, куда попадает текст, например, записан в базу данных или это имя файла (в ОС и файловых систем есть свои особенности).

**0**

Ответить

[[https://habrastorage.org/r/w48/getpro/habr/avatars/821/116/d67/821116d67a45444a79af0e2f0b3e1619.jpg](https://habr.com/ru/users/clipsa/)](https://habr.com/ru/users/clipsa/" \o "clipsa)

[**clipsa**](https://habr.com/ru/users/clipsa/)[9 июн в 11:08](https://habr.com/ru/articles/740026/#comment_25632332)

"**Парное тестирование**" в русскоязычном сообществе обычно называется "Попарное тестирование", т.к. "парное тестирование" - это как парное программирование, т.е. работа в паре, а попарное - это как раз когда мы рассматриваем пары значений.

Ну и туда же в попарное я б добавила в раздел "где использовать" что-то о том, что параметры должны быть взаимозависимы или взаимосвязаны, т.к. если " у вас большое количество входных параметров и большое количество возможных значений параметров", но эти параметры никак не связаны друг с другом, попарное тестирование будет только вредным.

# Как тестировать правильно: 8 техник тест-дизайна с примерами

[](https://highload.today/author/n-popelyshko-highload-today/)

[Наталия Попелышко](https://highload.today/author/n-popelyshko-highload-today/) BLOG

Senior Test Automation Engineer with .NET в SoftServe

Техники тест-дизайна — это правила и подходы, которые помогают создавать грамотные тест-кейсы. Они помогают нам тестировать, не просто переходя со страницы на страницу, а объясняют, почему мы вводим определенные значения и какие конкретно значения нужно вводить.

**Курс Fullstack Web Development.**

Володій навичками розробки як на фронтенді, так і на бекенді. За такими фахівцями черга в компаніях!

**[Отримати інформацію про курс](https://mate.academy/courses/fullstack?utm_source=highload&utm_medium=cpa" \t "_blank)**

И в этой статье мы поговорим про такие техники тест-дизайна:

* [Эквивалентное разделение (Equivalence Partitioning)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/" \l "1)
* [Граничные значения (Boundary Values)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#2)
* [Таблица принятия решений (Desicion Table)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#3)

**Курс Java.**

Всюди потрібні фахівці, які володіють магією програмування - особливо у світі Java! Приєднуйтесь і станьте частиною магії програмування!

**[Дізнатися про програму](https://mate.academy/courses/java?utm_source=highload&utm_medium=cpa" \t "_blank)**

* [Парное тестирование (Pairwise Testing)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#4)
* [Диаграмма перехода состояний (State-transition Diagram)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#5)
* [Диаграмма пользовательских ролей (Use Case Diagram)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#6)
* [Угадывание ошибок (Error Guessing)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#7)
* [Исследовательское тестирование (Exploratory Testing)](https://highload.today/blogs/8-tehnik-test-dizajna-s-primerami/#8)

Теперь — детально и с примерами про каждую из этих техник.

**Если вы хотите быстро получить необходимые знания для работы тестировщиком, то наши партнеры из**[**Robot Dreams**](https://robotdreams.cc/uk/course/qa-manual?utm_source=mctoday&utm_medium=partner&utm_campaign=sakhnenko_qam&utm_term=march_april)**и**[**Mate Academy**](https://mate.academy/courses/qa?utm_source=mc&utm_medium=cpa&utm_campaign=mct_partners)**с радостью вам помогут. Лучшие практикующие специалисты ждут вас.**

## Эквивалентное разделение

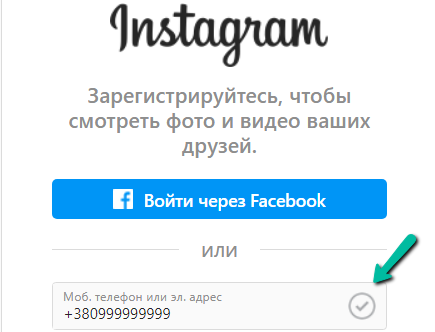
Помогать использовать первую технику будет сайт Instagram, а начнем с техники «Эквивалентное разделение». В чем ее суть? Мы берем все возможные варианты ввода текста и разделяем их на валидные и невалидные. Давайте протестируем поле «мобильный телефон» или «электронный адрес» — валидные и невалидные варианты ввода.

Введем +380999999999 — номер засчитан, как правильный, потому мы можем считать его валидным:

**Курс UI/UX Design.**

Закохайте своїх користувачів у ваш дизайн! Виглядайте стильно та перетворюйте їх у вірних прихильників!

**[Ознайомитись з курсом](https://mate.academy/courses/ui-ux?utm_source=highload&utm_medium=cpa" \t "_blank)**

[](https://highload.today/wp-content/uploads/2022/01/1-1.png)

Номер засчитан валидным

Теперь я хочу попробовать невалидный номер телефона и потому допишу сюда еще две цифры, и посмотрим, что скажет валидация (+38099999999999). Instagram показал нам крестик, то есть этот номер телефона уже не считается валидным, поскольку в нем много символов:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/2-3.png)

Пробуем дальше. Давайте я удалю два лишних символа и введу вместо одной цифры букву. Результат отрицательный, этот номер телефона тоже считается невалидным:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/3-1.png)

Теперь попробуем электронный адрес. Ввожу abc@gmail.com. Этот адрес посчитали неправильным, видимо потому что его уже используют:[тест дизайн](https://highload.today/wp-content/uploads/2022/01/4.png)

Пробуем такой — abcdfgty1234@gmail.com — он кажется уникальным, и после ввода видим, что Instagram посчитал его валидным. Делаем вывод, что невалидными считают те адреса, на которых люди уже зарегистрировались:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/5-2.png)

Попробуем еще один пример — natasha@gmail.com. По этому электронному адресу уже кто-то зарегистрирован, потому Instagram отнес его к невалидным:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/6-1.png)

Пробуем natash76554a@gmail.com — он валидный:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/7-2.png)

А что, если мы введем электронный адрес без символа «@»? Он считается неправильным:

**Курс QA.**

Стань неперевершеним QA спеціалістом! Стань технічною хрещеною феєю, яка оберігає від помилок і багів. Використай цей шанс!

**[Дізнатись про курс](https://mate.academy/courses/qa?utm_source=highload&utm_medium=cpa" \t "_blank)**

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/8.png)

А если вернуть символ собачки, но удалить точку? Он также считается невалидным:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/9.png)

Давайте попробуем поменять домен на другой. Например, natash76554a@ukr.net — он абсолютно правильный, и такая почта вряд ли существует:

[тест дизайн](https://highload.today/wp-content/uploads/2022/01/10.png)

Instagram тоже посчитал его верным. Итак, мы разделили все возможные вводы мобильного телефона или электронного адреса на валидные и невалидные группы. Вот результат:

|  |  |
| --- | --- |
| **Валидные** | **Невалидные** |
| +380999999999 | +38099999999999 |
| abcdfgty1234@gmail.com | +38099999999л |
| natash76554a@gmail.com | abc@gmail.com |
| natash76554a@ukr.net | natasha@gmail.com |
|  | natash76554agmail.com |
|  | natash76554a@gmailcom |

**Теперь можно составлять тест-кейсы.** Первым будет: зарегистрироваться с валидным номером телефона, и конкретно этот номер телефона можно использовать как тестовые данные. Второй — зарегистрироваться с валидным электронным ящиком и использовать какой-то из примеров или же сразу все ящики.

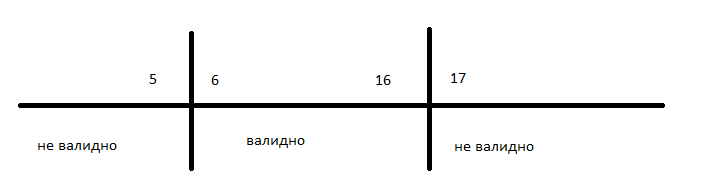
**К невалидным будут относиться:**

* использовать номер телефона с большим количеством символов;
* использовать номер телефона с буквами вместо цифр;
* использовать электронный ящик, который уже зарегистрирован в Instagram;
* использовать электронный ящик без собачки, без точки, с несуществующим доменом.

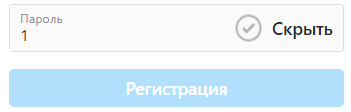
## Граничные значения

Теперь я покажу, как использовать технику «Граничных значений». В этой технике мы работаем только с цифрами. Допустим, у нас есть требование, что пароль должен быть от 6 до 16 символов. То есть на границе от 6 до 16 это должны быть четыре числа:

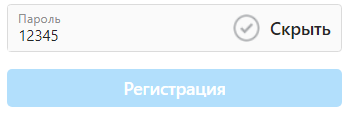
* 5 — потому что 5 не входит в границу;
* 6 — потому что с шести символов можно начинать;
* 16 — потому что 16 символов — это максимум;
* 17 — потому что это больше максимума.

[](https://highload.today/wp-content/uploads/2022/01/11-2.png)

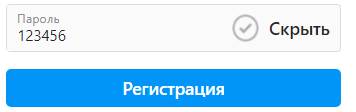
Давайте тестировать. Сначала я хочу ради эксперимента ввести пароль с одним символом. Кнопка регистрации серая — то есть с одним символом невозможно:

[](https://highload.today/wp-content/uploads/2022/01/12-1.png)

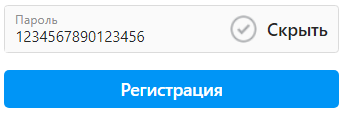
Вводим пароль из пять символов — кнопка регистрации все еще серая:

[](https://highload.today/wp-content/uploads/2022/01/13.png)

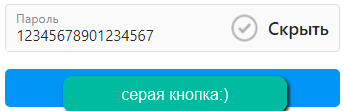
Вводим пароль из 6 символов — кнопка регистрации засветилась, то есть регистрация позволена:

[](https://highload.today/wp-content/uploads/2022/01/14-1.png)

И пароль из 16 символов — кнопка все еще голубая:

[](https://highload.today/wp-content/uploads/2022/01/15-1.png)

Ну и поскольку в Instagram нет ограничений по количеству символов в пароле (или мне лень вводить 256 символов :)), то сделаем вид, что 17 — это максимум. Ввожу 17 символов и мы делаем вид, что кнопка «Зарегистрироваться» стала серой:

[](https://highload.today/wp-content/uploads/2022/01/16-1.png)

Давайте разделим эти варианты ввода на валидные и невалидные значения:

|  |  |
| --- | --- |
| **Валидные** | **Невалидные** |
| 123456 | 12345 |
| 1234567890123456 | 12345678901234567 |

В **валидные** пойдут: пароль из 6-ти символов и пароль из 16-ти символов. В **невалидные** — пароль из 5-ти символов и пароль из 17-ти символов. То есть, чтобы использовать в тест-кейсе граничные значения, нам нужно использовать четыре шага: два из них — это граничные валидные значения (6 и 16) и вторые два — это граничные невалидные значения, то есть 5 и 17.

Где еще можно использовать эту технику тест-дизайна? Везде, где фигурируют цифры: **например, возраст.** Давайте представим, что есть сайт, на котором регистрация позволена людям от 18 до 60 лет. И посчитаем, какие значения можно и нельзя вводить. Итак, валидные значения — это 18 и 60. Тогда невалидным значением будет 17, потому что в 17 лет нас не должно зарегистрировать, и 61, потому что после 60-ти тоже не должно регистрировать по правилам выдуманного нами сайта:

|  |  |
| --- | --- |
| **Валидные** | **Невалидные** |
| 18 | 17 |
| 60 | 61 |

Где еще может использоваться эта техника? **Например, в номере телефона.** Все мы знаем, что валидный номер телефона состоит из 10 цифр, которые не относятся к коду страны ((+38)099 999 99 99). Это будет валидный номер телефона. Но что, если я хочу создать номер телефона из 11 цифр без кода страны (или из 9)? Номер телефона считается невалидным в таких случаях:

|  |  |
| --- | --- |
| **Валидные** | **Невалидные** |
| +38 099 999 99 99 | +38 099 999 99 99 9 |
|  | +38 099 999 99 9 |

### Какие тест-кейсы мы могли создать из упомянутых примеров?

1. Создать пароль из 6-ти символов.

2. Создать пароль из 16-ти символов.

3. Создать пароль из 5-ти символов и ожидать, что регистрация не пройдет.

4. Создать пароль из 17-ти символов и ожидать, что регистрация не пройдет.

5. Зарегистрироваться с возрастом 18 лет.

6. Зарегистрироваться с возрастом 60 лет.

7. Зарегистрироваться с возрастом 17 лет.

8. Зарегистрироваться с возрастом 61 год.

9. Зарегистрироваться, указав правильный номер телефона из 10 символов без кода страны.

10. Зарегистрироваться, указав неправильный номер телефона из 9 символов без кода страны.

11. Зарегистрироваться, указав неправильный номер телефона из 11 символов без кода страны.

### Зачем нужна эта техника?

**Для того, чтобы люди не тестировали 10 символов в пароле, которые никому не нужны, потому что они находятся на границе между 6 и 16.** Для того, чтобы не тестировать возраст 20, 30, 40, потому что все эти значения находятся между 18 и 60. Также для того, чтобы люди не тестировали возраст 100 лет или -100 лет. Для того, чтоб не проверять номер телефона из 1, 2, 100 цифр. Потому что все эти проверки будут бессмысленными, ведь мы уже протестировали граничные значения.

## Таблица принятия решений

Следующая техника тест-дизайна — «Таблица принятия решений» или ТПР.

Нужно нарисовать таблицу, в которой мы будем использовать разные условия и ситуации.

На примере Instagram мы должны заполнить четыре поля:

* мобильный телефон или электронный адрес;
* имя и фамилию;
* имя пользователя (ник);
* пароль.

И мы будем решать, валидные или невалидные ситуации мы ввели, и показывать ожидаемый результат. Но сначала давайте условимся, как мы будем обозначать валидные и невалидные способы ввода:

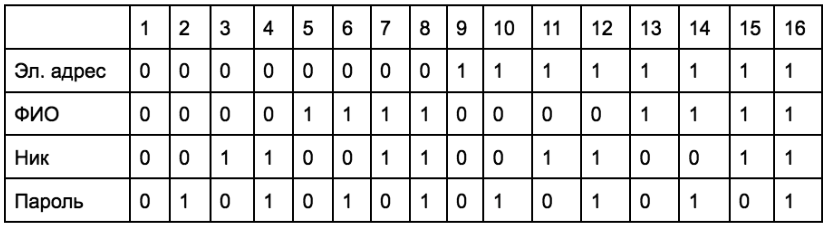
1. **Валидные** мы будем обозначать словом true либо буквой t, либо цифрой 1.
2. **Невалидные** мы будем обозначать либо словом false либо буквой f, либо цифрой 0.

Почему цифры 1 и 0? Так пишут программисты. В бинарном коде: 0 — это ложь, 1 — это правда. Например, если лампочка горит — это 1, а если она выключена — 0.

Теперь нам нужно создать таблицу. Это делается с помощью дискретной математики.

Рисуем таблицу на 16 колонок. 16 — это если возвести 2 в 4 (а столько у нас полей) степень, то выйдет 16.

Нам нужно сделать так, чтобы все варианты ввода отличались друг от друга, то есть чтобы не было двух одинаковых, как я сделала в этом примере:



Поскольку у нас 16 колонок, то делим 16 на 2 и получаем 8. В первой строке этой таблицы первые 8 цифр заполняем нолями, вторые 8 цифр — единицами.

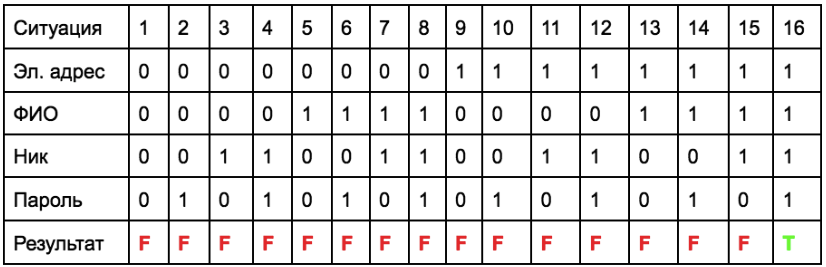
Чтоб заполнить следующую строку, делим уже 8 на 2 и получаем 4. Заполняем так: 4 ноля, 4 единицы, 4 ноля, 4 единицы.

Для следующей строки делим 4 на 2. Получается 2 ноля, 2 единицы, 2 ноля, 2 единицы, 2 ноля, 2 единицы, 2 ноля, 2 единицы.

И в последней строке проставляем 0 и 1 через 1: 0, 1, 0, 1, 0 , 1…

С помощью этой тактики мы получили уникальные колонки, то есть нет ни одной одинаковой пары.

Теперь давайте использовать эту таблицу:



В **первой ситуации (столбце)**, мы вводим невалидный электронный ящик, имя пользователя, имя и фамилию, пароль. Если попробовать ввести все эти данные невалидно, то увидим, что регистрироваться нам не дадут, поэтому в строку с результатом вносим букву F — fail, и для наглядности можем сделать эту клеточку красной.

Так я буду проходиться по каждой колонке этой таблицы, пока не заполню все 16 результатов.

Во **второй строке** я должна ввести все невалидные данные, кроме пароля. Пробую на практике ввести нормальный пароль и нажать «Регистрация» и получаю два сообщения об ошибке:

* Это имя пользователя уже занято.
* Неправильный номер телефона.

В результат в таблице тоже пишу F, а в идеале в ту клеточку нужно вставить оба ожидаемых сообщения об ошибке.

Продолжаем пробовать все ситуации до конца таблицы. Конкретно в этом примере все результаты, кроме последней колонки, имеют статус Failed, а последняя — Pass, поскольку там мы ввели правильный мобильный телефон, официальное имя, никнейм и пароль.

**Давайте создадим еще одну таблицу, но попроще, для примера.**

Допустим, нам нужно загрузить аватарку, и есть требования: она должна весить меньше 1 Гб и быть в формате фото (jpg, png или gif). В таблице я делаю две строки:

* Фото ли это?
* Валидный ли размер?

Два во второй степени равно четырем, так что у нас будет всего четыре колонки:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **1** | **2** | **3** | **4** |
| **Формат фото?** | 0 | 0 | 1 | 1 |
| **Размер меньше 1 Гб?** | 0 | 1 | 0 | 1 |
| **Результат** | Fail | Fail | Fail | Pass |

**Пройдемся по каждой колонке:**

1. Если у нас не фотография (а, например, mp3) и размер больше 1 Гб, то результат будет Fail.
2. Если мы загрузим не фотографию, но меньше 1 Гб, то все равно Fail.
3. Если мы загрузим фотографию, но больше 1 Гб, то результат тоже Fail.
4. Если мы загружаем фото с правильным форматом и меньше 1 Гб, то результат Pass.

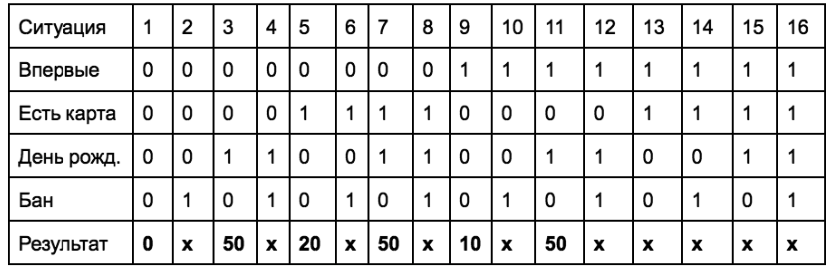
А теперь я покажу еще один пример, только он будет интересней. Тут результат будет не Fail и Pass, а цифры. Но в клеточках таблицы также будут ноли и единицы.

Представьте себе, что вы пошли в супермаркет и для привлечения покупателей владелец придумал систему скидок:

* Если ты впервые в магазине — у тебя 10% скидки.
* Если ты постоянный клиент — то у тебя есть золотая карта и 20% скидки.
* Если у тебя сегодня день рождения — то на все покупки тебе дают 50% скидки.
* Если ты, допустим, разбил там бутылку вина и убежал, то тебя кидают в бан и ничего не будут продавать.

Теперь давайте рассмотрим все эти ситуации и их результат:

1. Клиент не новенький, нет карты, нет дня рождения и в бане он не состоит: результат — 0% скидки.
2. Клиент не новенький, нет карты, нет дня рождения, но в бане — его выгонят, ставим на нем Х.
3. Клиент не новенький, нет карты, у него день рождения и он не в бане — 50%.
4. Клиент не новенький, нет карты, у него день рождения и он в бане — выгоняем.
5. Клиент не новенький, у него есть карта на скидку, нет дня рождения и в бане он не состоит — 20%.
6. Клиент не новенький, у него есть карта на скидку, нет дня рождения и он в бане — выгоняем.
7. Клиент не новенький, у него есть карта на скидку и день рождения, в бане он не состоит — тут надо спрашивать владельца магазина: прибавляем скидку или берем большую (70% или 50%?).
8. Клиент не новенький, у него есть карта на скидку и день рождения, но он в бане — выгоняем.
9. Клиент новенький, нет карты, нет дня рождения и в бане он не состоит — 10% скидки.
10. Клиент новенький, нет карты, нет дня рождения, но в бане — его выгонят.
11. Клиент новенький, нет карты, у него день рождения и он не в бане — решаем: 60% или 50%.
12. Клиент новенький, нет карты, у него день рождения и он в бане — выгоняем.
13. Клиент новенький, у него есть карта на скидку, нет дня рождения и в бане он не состоит — значит, он нас обманывает, и карта не его — кидаем в бан 🙂
14. Клиент новенький, у него есть карта на скидку, нет дня рождения и он в бане — выгоняем.
15. Клиент новенький, у него есть карта на скидку и день рождения, в бане он не состоит — значит он нас пытается обмануть — выгоняем.
16. Клиент новенький, у него есть карта на скидку и день рождения, но он в бане — выгоняем.



С помощью этой таблицы мы увидели интересные вещи, которые точно пропустили бы без нее.

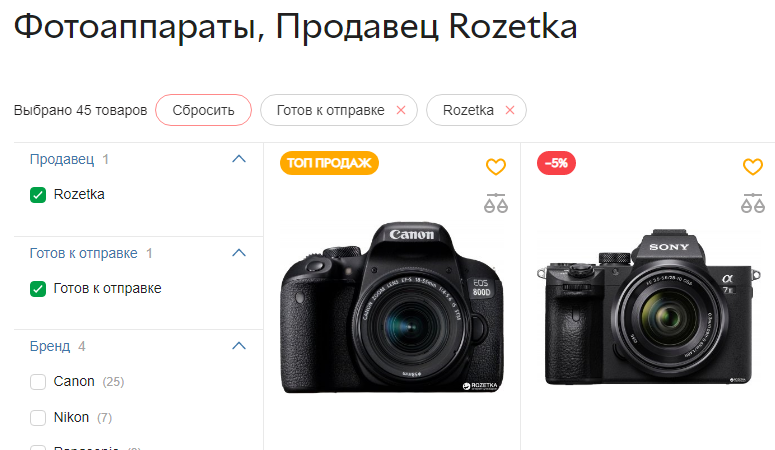
Например, что делать, если у нас есть две скидки (20% и 50%) одновременно: прибавлять их или использовать большую из них? Или такая ситуация: если клиент новенький, но пришел со скидочной картой, значит он ее у кого-то взял и пытается обмануть магазин. Что мы делаем в такой ситуации: прощаем и используем карточную скидку или выгоняем?

Соответственно, можно создать целых 16 интересных и уникальных тест-кейсов для 16 разных ситуаций. Помочь разобраться во всех аспектах помогут специалисты [Robot Dreams](https://robotdreams.cc/uk/course/qa-manual?utm_source=mctoday&utm_medium=partner&utm_campaign=sakhnenko_qam&utm_term=march_april) со своим курсом QA Manual.

## Парное тестирование

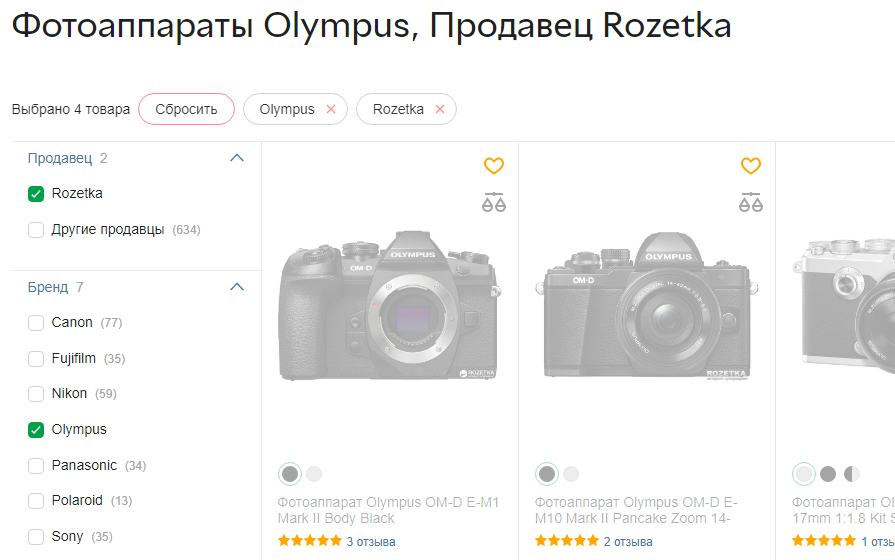
Для техники «Парное тестирование» нужно открыть любой интернет-магазин и каталог товаров. Пусть это будут «Cмартфоны, ТВ и электроника».

Давайте посмотрим на левую панель. Здесь есть обширный фильтр: по продавцу, отправке, бренду, алфавиту, цене, беспроцентному кредиту и т.д. В чем суть парного тестирования? Мы имеем много разных характеристик, по которым нужно сортировать, но мы будем тестировать не по одной характеристике, а сразу по двум. Для чего? Чтобы проверить, какая будет реакция у системы, какой будет результат. Переходим к практике. Сначала мы должны протестировать продавца и готовность к отправке. Ставим две галочки:

[](https://highload.today/wp-content/uploads/2022/01/17.png)

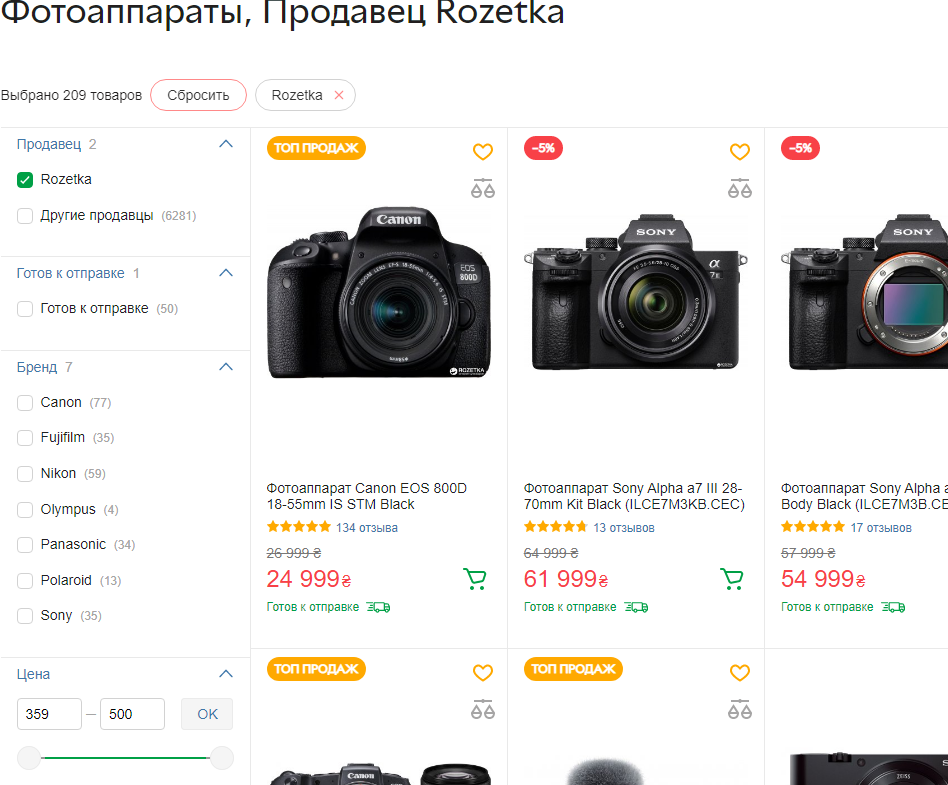
Ставим две галочки в фильтрах

И проверяем, что есть товары, отфильтрованные по продавцу и по готовности к отправке. Теперь убираем галочку с «Готов к отправке» и ставим следующий фильтр, например, бренд:

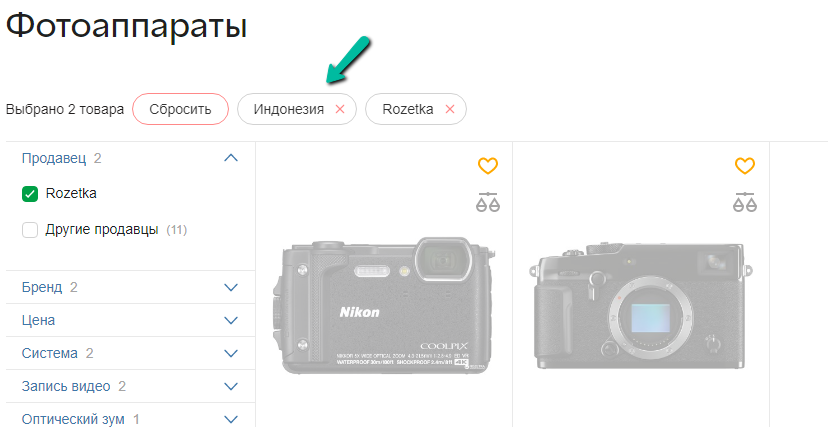
[](https://highload.today/wp-content/uploads/2022/01/18-1.png)

Применяем следующий фильтр

То есть фильтруем по продавцу и по бренду, и проверяем, есть ли результаты. Результат есть — значит эта пара прошла. Идем дальше, бренд анчекаем, фильтруем по продавцу и по цене: выставляем цену от 359 до 500 грн. И этот фильтр тоже работает — тест прошел.

[](https://highload.today/wp-content/uploads/2022/01/19-1.png)

Идем дальше, убираем фильтр цены и ставим на «страну производителя», допустим, Индонезия. Видим, что отфильтровано два товара — тест пройден:

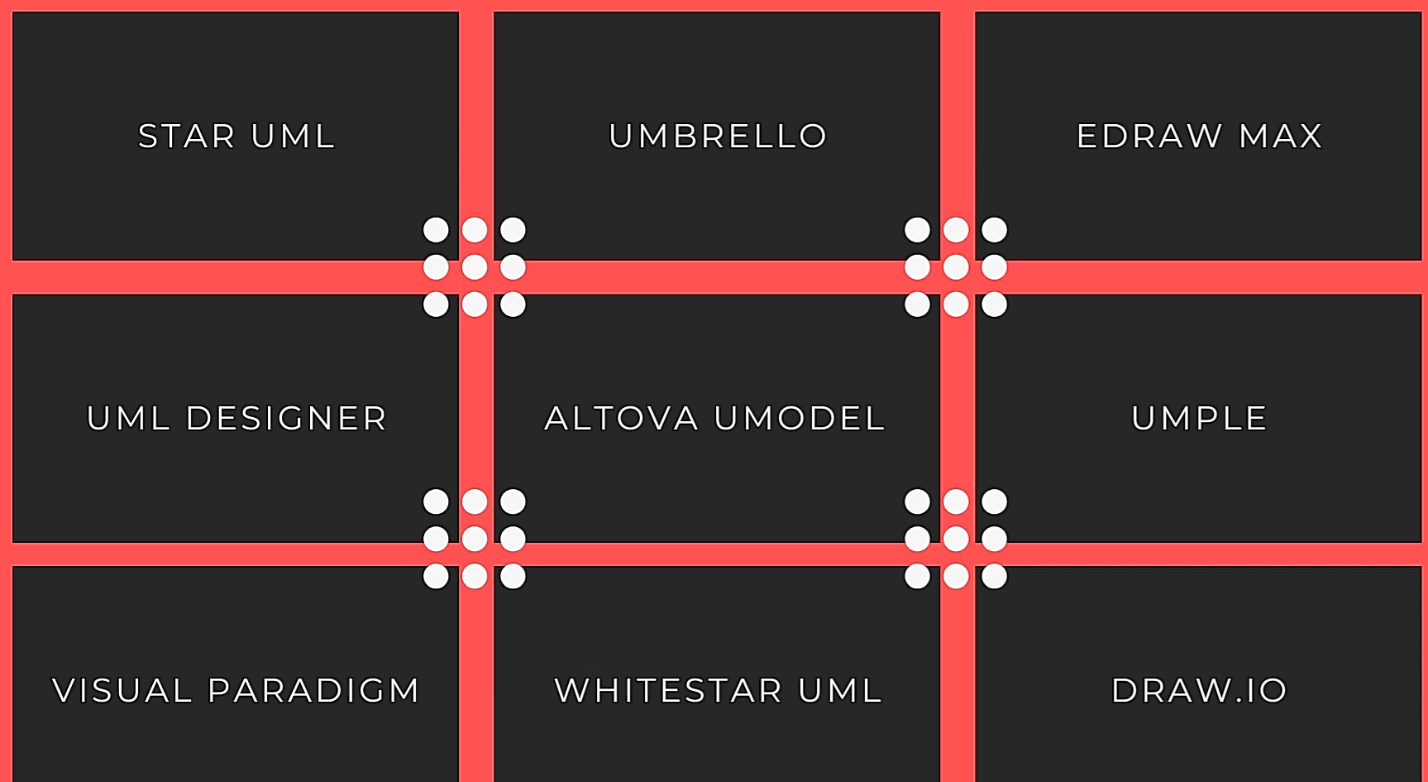
[](https://highload.today/wp-content/uploads/2022/01/20-1.png)

Добавляем страну-производителя

Убираем страну-производителя и ставим фильтр на страну регистрации бренда — Австралию. И так проходимся по комбинациям из двух разных фильтров, пока не протестируем все со всеми (первый и второй, первый и третий, второй и третий).

## Диаграмма перехода состояний

Следующая техника тест-дизайна — работа с диаграммой. И перед тем, как начинать, было бы логично показать, на каких сайтах и с помощью каких программ их удобно рисовать:

[](https://highload.today/wp-content/uploads/2022/01/21.png)

Сайты и программы, с помощью которых удобно рисовать диаграммы

Первая диаграмма, которую мы рассмотрим — «Диаграмма перехода состояний» (State Transition Diagram). Всегда нужно учить английские термины, потому что спрашивают именно их. Итак, из чего состоит диаграмма перехода состояний? Из кружочков и стрелочек 🙂

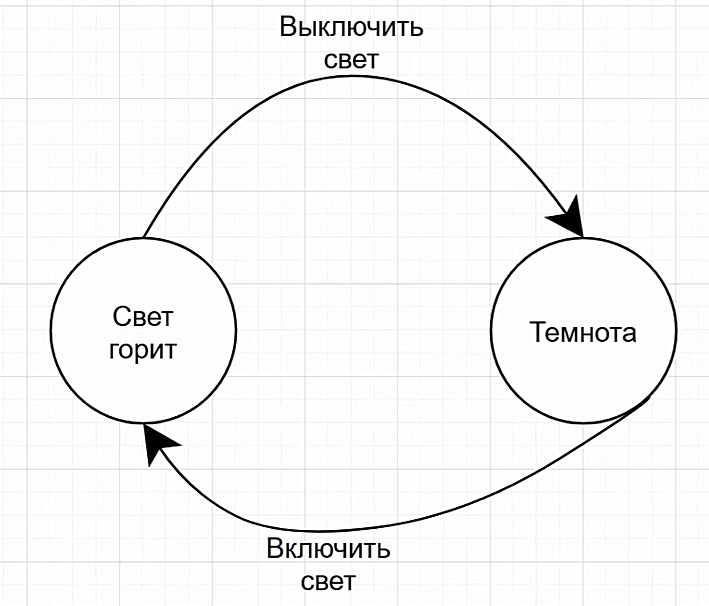
[](https://highload.today/wp-content/uploads/2022/01/22.png)

Диаграмма перехода состояний (State Transition Diagram)

Что у нас получилось? Это самый элементарный пример диаграммы перехода состояний, а именно — пример выключателя света: если мы выключим — то свет не горит, если включим— свет горит. Над стрелочками принято писать действия, которые мы делаем, например, «выключить свет» и «включить свет». А внутри кружочка пишутся состояния, которые в этот момент происходят. Например, если мы выключим свет, то состояние будет «темнота», а если включим, то состояние «светло».

**Теперь давайте рассмотрим более сложную систему, например систему входа в социальную сеть.**

Говорят, что каждая диаграмма перехода состояний должна иметь начало и конец, по этому мы рисуем кружочки и для этих состояний:

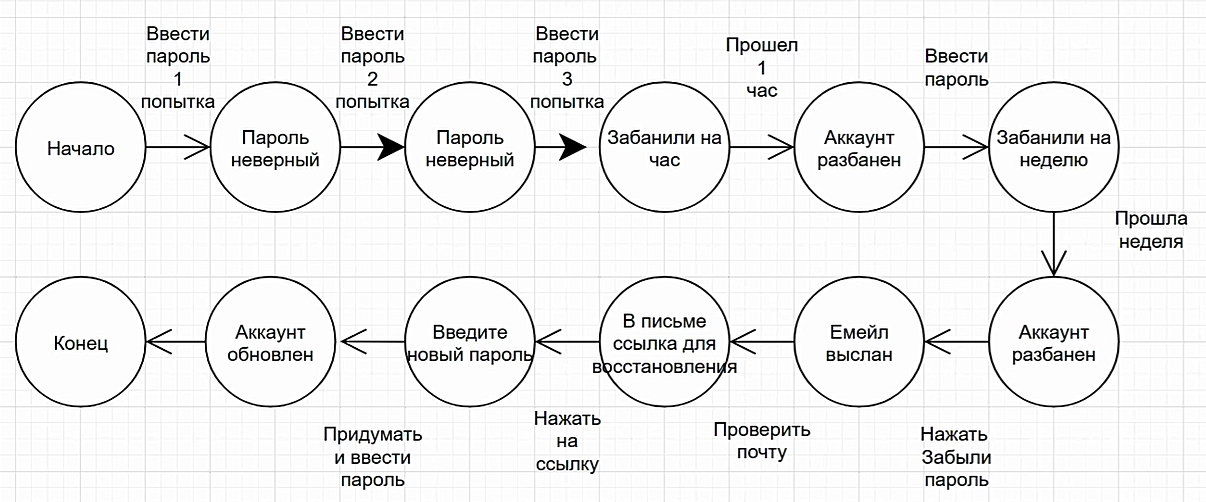
[](https://highload.today/wp-content/uploads/2022/01/23.png)

Диаграмма системы входа в социальную сеть

Сначала пользователь просто заходит на сайт и вводит логин и неправильный пароль (ведь мы его забыли :)). Результат — пароль неправильный.

Дальше вводим пароль со второй попытки (и опять неправильный). Результат — пароль снова неверный.

Третья попытка — снова вводим неверный пароль. Результат — пароль снова неверный и пользователя забанят на час.

Следующее действие — ждем, пока пройдет час. Результат — аккаунт разбанен.

Что дальше? Давайте снова введем неверный пароль. Поскольку система помнит, что у нас было три неудачных попытки, то она забанит нас на неделю.

Ждем неделю. Результат — аккаунт разбанен.

Давайте мы наконец-то поумнеем и нажмем на кнопку «Забыли пароль». Результат — на электронную почту приходит письмо для восстановления. Дальше пользователь проверяет почту, и как результат — в письме пришла ссылка для восстановления пароля.

Пользователь нажимает на ссылку для восстановления пароля, и видит сообщение «Введите новый пароль». Пользователь вводит пароль и аккаунт обновляется.

Зачем нужны эти диаграммы? Если у нас есть море вариантов, то мы легко можем запутаться или не покрыть все возможные варианты тест-кейсами.

## Диаграмма пользовательских ролей

Следующая техника будет называться «Диаграмма пользовательских ролей» (Use Case Diagram).

Представим себе обычный интернет-магазин. Какие там есть роли? Например:

* Администратор
* Продавец
* Покупатель (зарегистрированный)
* Незарегистрированный пользователь

У каждой роли есть свои права и доступы, и каждый из них умеет что-то делать. Поэтому рисуем четырех человечков, подписываем их роли и переходим к их действиям. Все действия будем указывать в небольших прямоугольниках. Проведем стрелочку от роли к доступному действию.

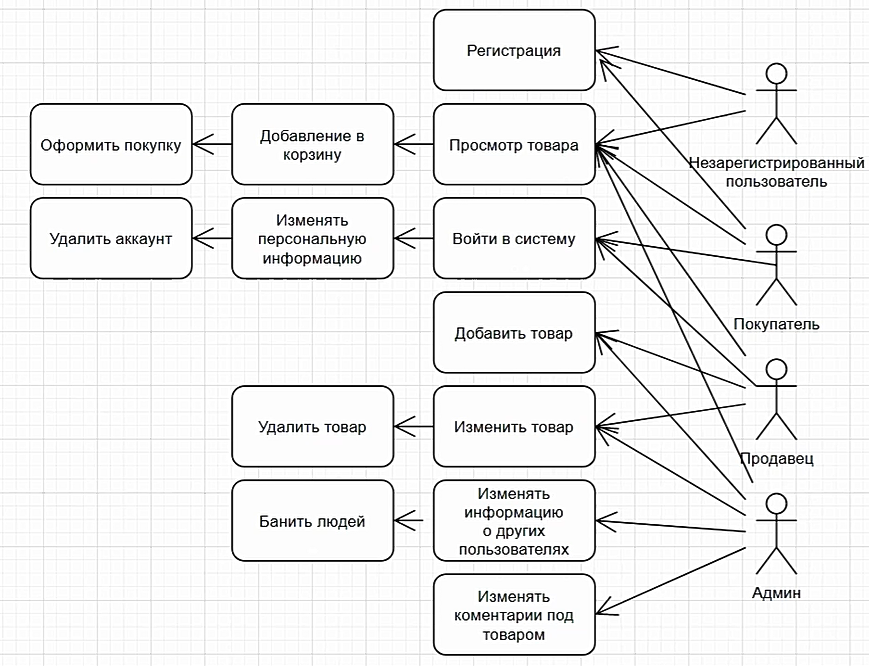
[](https://highload.today/wp-content/uploads/2022/01/24.png)

Диаграмма ролей и действий в интернет-магазине

Давайте начнем с **незарегистрированного пользователя.** Он умеет:

* Регистрироваться
* Смотреть товар
* Добавлять товар в корзину
* Оформлять покупку

Рисуем от незарегистрированного пользователя стрелочки к первым двум действиям. Чтоб добавить товар в корзину, его нужно открыть, то есть мы не можем кинуть товар в корзину, не посмотрев на него. Поэтому к действию «Добавлять товар в корзину» рисуем стрелочку от «Смотреть товар», а не от роли. То же самое с оформлением покупки, мы не можем оформить ее, не добавив товар в корзину, потому ведем стрелочку от «Добавить товар в корзину». У нас получилась небольшая цепочка с правами.

Теперь давайте посмотрим, что может **покупатель**:

* Все действия с предыдущей роли
* Войти в систему (залогиниться)
* Изменять персональную информацию (только если вошли в систему, потому стрелочка от пункта 2)
* Удалить аккаунт (со страницы изменения информации, потому стрелочка от пункта 3)

Что может делать **продавец**:

* Все действия ролей выше (кроме регистрации :))
* Добавлять товар
* Изменить товар
* Удалить товар (со страницы изменения, стрелочка из пункта 3)

Что может **администратор**:

* Изменять информацию о других пользователях
* Добавлять людей в бан (с пункта 1)
* Удалять и изменять комментарии
* Все действия ролей выше (кроме регистрации)

Кроме очевидных тест-кейсов, которые приходят всем на ум, глядя на эту диаграмму, можно протестировать такие вещи:

1. Незарегистрированный пользователь не может изменять информацию о товаре

2. Незарегистрированный пользователь не может банить людей

3. Продавец не может банить людей

4. Администратор не может регистрироваться

Все предыдущие техники назывались техниками «черного ящика». Есть еще техники «белого ящика», но для них надо знать языки программирования и работать с кодом. Потому мы поговорим про следующие техники тест-дизайна, которые относятся к отдельной категории: тестирование, связанное с опытом.

## Угадывание ошибок

Начнем с угадывания ошибок (Error Guessing). В этой технике нужны опытные ребята, которые могут придумать и вспомнить ситуации, в которых ПО «ломается». Обычно эти ситуации встречались им с предыдущим опытом. Именно эта техника сильно зависит от мастерства, ведь только опытные специалисты знают, где искать баг.

Какие типичные условия следует попробовать, чтоб пройти угадывание ошибок:

* Уменьшать ширину окна по чуть-чуть, чтоб отловить баги в User Interface
* Делить на ноль
* Ввести пустое значение в поле
* Ввести пробел в начало, в середину, в конец поля
* Негативное тестирование дат (30, 31 февраля, а также даты, которые еще не наступили)
* Ставить на аватарку текстовые файлы, музыку, пустые файлы

Это самые часто встречающиеся ошибки.

## Исследовательское тестирование

Переходим к последней технике. Это исследовательское тестирование.

Его нужно применять в ситуациях:

1. Когда мы не знаем продукт. Мы изучаем его, как обычный пользователь. Например, нам дали новый проект и его нужно изучить: прокликать и узнать, какой там функционал.
2. Когда нам не нужна или у нас нет документации. Мы используем исследовательское тестирование, когда нет времени писать тест-кейсы или сроки горят, или нам приказали не писать документацию 🙂
3. Когда нет времени. Если дают сайт или программу и говорят (образно), что есть 15 минут, чтоб найти 100 багов и нет времени ни создавать тест-кейсы, ни писать репорты, тебе просто нужно срочно тестировать и проверять качество — это исследовательское тестирование.

Окей, мы изучили, что это тестирование без подготовки и без документации, тогда назревает вопрос: а **в чем разница между исследовательским и свободным тестированием?** Для свободного нужно меньше мастерства, его может провести и девелопер, а для исследовательского нужно владение техниками (и не только тест-дизайна). Для того чтобы обрести навык работы с разными техниками нужен [хороший ментор](https://mate.academy/courses/qa?utm_source=mc&utm_medium=cpa&utm_campaign=mct_partners), который сможет ответить на ваши вопросы и подсказать оптимальное решение.