

# Рассмотрим темы:

---

- ✓ Методы исследования продукта
- ✓ Оформление тестов (тест-кейсы, чек листы)
- ✓ Техники тест дизайна
  - Классы эквивалентности
  - Граничные значения
  - Таблица решений
  - Диаграмма состояний
  - Сценарии использования
  - Парное тестирование

# Исследование продукта

---

Мы устроились на работу, и нам дали задачу протестировать уже готовый продукт.

Что делать?

- ✓ исследовать продукт - что он умеет;
- ✓ правильно задавать вопросы;
- ✓ использовать инструменты для записи результатов.

- ✓ Спецификация есть – изучаем ее
- ✓ Спецификации нет – задаем вопросы



# Как правильно задавать вопросы

---

Вопросы бывают открытые и закрытые .

<b>Закрытые вопросы</b>	<b>Открытые вопросы</b>
<p><i>Возможен только однозначный ответ:</i></p> <p>дата; время; название; количество; ДА или НЕТ? Например: Сколько лет вы работаете в тестировании? Вы проводили нагрузочное тестирование? Система должна обрабатывать TXT-файлы?</p>	<p><i>Нужен развернутый ответ</i></p> <p>Открытые вопросы обычно начинаются со слов:</p> <p>Что? Как? Почему? Каким образом? При каких условиях?</p>

**Фактически закрытый вопрос означает, что вы сами уже додумали себе требования и их же и будете тестировать. А тестировщик НИКОГДА не додумывает. Ведь иначе он будет тестировать вообще не то, что нужно.**

Например, задача «**система принимает на входе файл с текстом и проверяет его на опечатки**». Тестировщик додумал себе, что это простой блокнот, и уточняет у заказчика:

- Система должна обрабатывать TXT -файлы?
- Да, должна.
- Ок, спасибо.

Полностью уверенный в том, что узнал всё, что ему было нужно, тестировщик готовит свои тесты. А на самом деле заказчик ждет, что система будет уметь работать со всеми текстовыми форматами, включая Excel, **а еще она сможет считывать текст с картинок! И именно в этом ее фишка**. То есть главный функционал. А тестировщик картинки с текстом вообще не проверяет, так как считает это негативным тестированием.

- ✓ Всегда старайтесь задавать максимально открытый вопрос.
- ✓ Если видите, что ваш вопрос «закрытый», подумайте, что именно вы хотели узнать, -об этом и спросите.

# Как оформить результат, чтобы не забыть всё, что мы узнали?

---

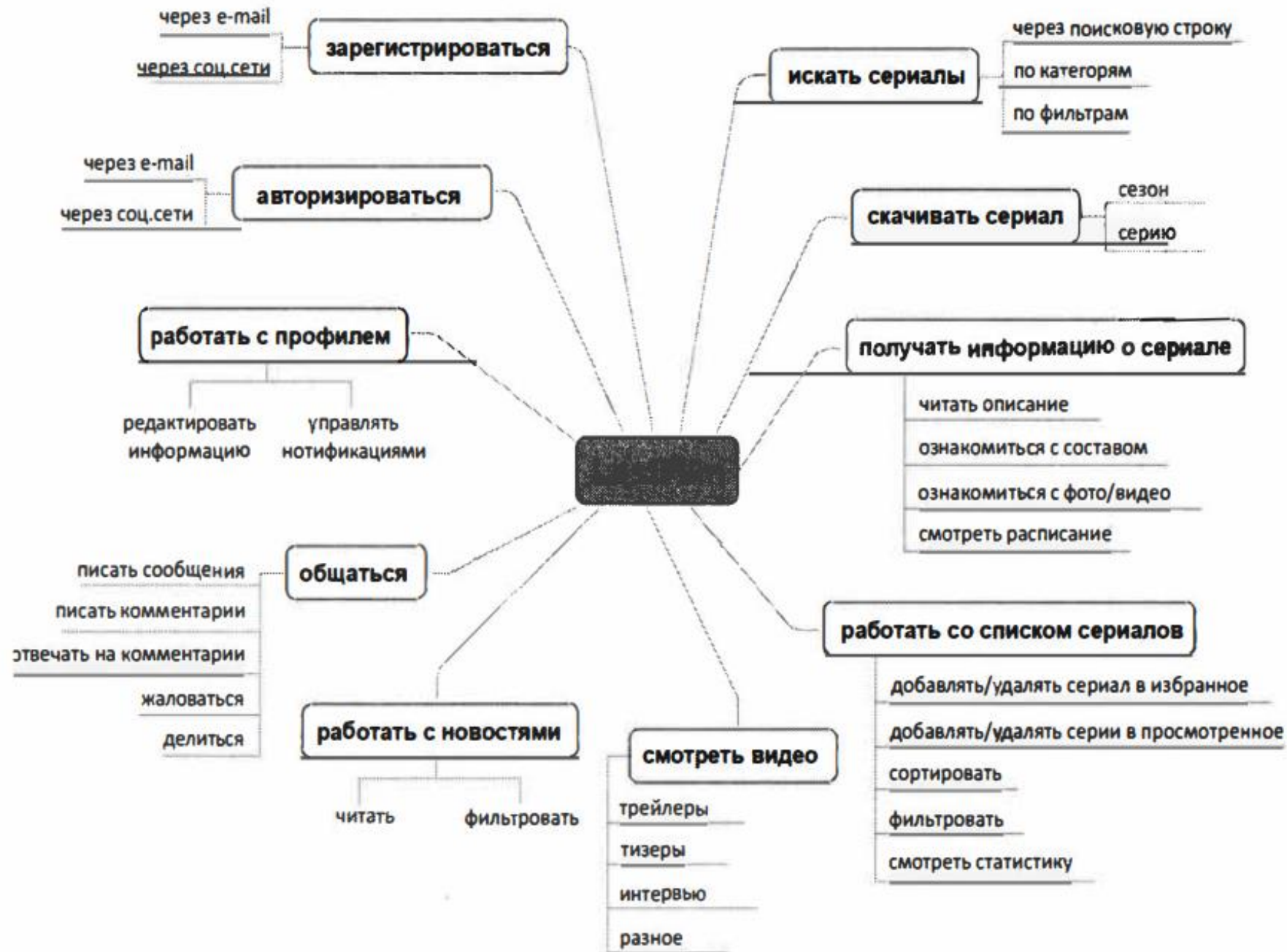
- ✓ Записать в блокнот
- ✓ Нарисовать карту приложения

## Как нарисовать карту приложения?

*Есть несколько основных правил, которые помогут сделать карту простой и понятной.*

Выделите основные функции, задав себе вопросы:

- ❖ *зачем пользователю наш продукт?*
- ❖ *что он там делает?*
- ❖ *что мы хотим, чтобы он делал?*



По важности

Регистрация  
здесь не самая  
важная функция,  
поэтому она в  
конце

# После изучения функционала продукта

---

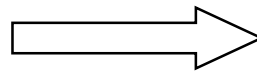
- проектируем тесты;
- оформляем тест-кейсы;
- пишем чек-листы.

## Проектирование тестов

*При продумывании тестов нужно помнить о том, каким должен быть тест и о приоритетах выполнения.*

### Каким должен быть тест?

- Тест должен быть конкретным
- У теста должен быть результат



Из описания теста должно быть понятно:

- что сделать;
- как сделать.

### Приоритеты выполнения тестов

ВСЕГДА начинаем с позитивного тестирования и только потом проводим негативное. Потому что задача тестировщика - не найти как можно больше багов, а дать наиболее полную информацию по приложению



# Как оформлять тесты?

---

## 1. Тест-кейс (ТК) - это подробное описание проверки.

ТК1. Авторизация через email

### **Предварительные шаги**

Зарегистрироваться с email test@test.ru и паролем 1

### **Шаги**

1. Открыть сайт <https://www.example.com/>.
2. Нажать на кнопку «Вход» в правом верхнем углу окна сайта.
3. В поле «Email» ввести значение: test@test.ru.
4. В поле «Пароль» ввести значение: 1.
5. Нажать кнопку «Войти».

### **Ожидаемый результат**

- Открылась главная страница сайта.
- В верхнем правом углу страницы отображается значок личного кабинета (недоступен без авторизации)

# Структура тест-кейса

---

1. шаги — это инструкция по вводу;
2. исполнение шагов — это ввод;
3. ожидаемый результат — это ожидаемый вывод;
4. фактический результат — это фактический вывод.

**Исполнение тест-кейса завершается сравнением вывода фактического и вывода ожидаемого.**

1. Положительный исход (PASS), если ФР равен ОР,
2. Отрицательный исход (FAIL), если ФР не равен ОР: найден баг!

Иногда возникает ситуация, когда мы заблокированы, так как не можем пройти ВСЕ шаги тест-кейса. Например, мы не можем продвинуться дальше, если кнопки "Завершить заказ" из шага 14 не существует на соответствующей веб-странице. В таком случае мы рапортуем баг (в данном случае баг об отсутствии кнопки "Завершить заказ") и откладываем исполнение тест-кейса до устранения бага.

# Атрибуты тест-кейса (рекомендуемые)

---

- **Уникальный идентификатор тест-кейса.**
- **Приоритет** — важность по шкале от 1 (высший приоритет) до n (низший приоритет).
- **Название** — основная тема, или идея тест-кейса. Краткое описание его сути.
- **Предусловия** — описание условий, которые не имеют прямого отношения к проверяемому функционалу, но должны быть выполнены. Например, «пользователь авторизован».
- **Постусловия** — действия, которые нужно сделать после проведения проверки. Например, удалить внесённые данные из БД.
- **Окружение** — на каком устройстве, в каком браузере. Иногда заполняют до тестирования, иногда — сам тестирующий указывает после.
- **История редактирования** — кто создал/изменил? Когда? Зачем? Почему?
- **Шаги** — описание последовательности действий, которая должна привести нас к ожидаемому результату
- **Ожидаемый результат** — что мы ожидаем увидеть после выполнения шагов.

# Пример тест-кейса

QA Tester's Log		Review comments from Bill incorporate in version 2.1							
Tester's Name		Mark	Date Tested		1-Jan-2017	Test Case (Pass/Fail/Not		Pass	
S #	Prerequisites:			S #	Test Data				
1	Access to Chrome Browser			1	Userid = mg12345				
2				2	Pass = df12@434c				
3				3					
4				4					
Test Scenario		Verify on entering valid userid and password, the customer can login							
Step #	Step Details		Expected Results		Actual Results		Pass / Fail / Not executed / Suspended		
1	Navigate to http://demo.guru99.com		Site should open		As Expected		Pass		
2	Enter Userid & Password		Credential can be entered		As Expected		Pass		
3	Click Submit		Cutomer is logged in		As Expected		Pass		
4									

## Особенности тест-кейсов

- **В них максимум информации.** Их легко можно отдать новичку, и он проведет проверки, не задавая вопросов.
- **Они независимые.** Хорошие тест-кейсы не зависят друг от друга, поэтому можно легко редактировать один, не боясь, что развалятся все остальные.
- **Они небольшие.** Один тест-кейс – одна проверка.

### Тест-кейсы нужны, когда у нас:

- **жизненно важные системы**, ошибка в которых может привести к гибели людей (авиастроение, медицина, ПО для атомных станций). Надо тестировать очень тщательно;
- **сложная система или сложная часть системы.** Чтобы каждый раз не вспоминать «а как мне это сделать?», лучше написать тест-кейс.
- **в команде постоянно появляются новые люди.**

### Тест-кейсы не нужны, когда у нас:

- **простые системы** (веб-сайты одностраничники, мобильные приложения и т. п.);
- **если в команде всего один или два тестировщика, знающие свой продукт.** Время, потраченное на создание и поддержку тест-кейсов, никогда не окупится.

Так как тест-кейсы очень сложно поддерживать, то чаще используют чек-листы или комбинацию «чек-листы & тест-кейсы».

В последнем случае большинство проверок пишут в виде чек-листов, а особо сложные уже в виде тест-кейсов, чтобы каждый раз не вспоминать, как этот хитрый сценарий работает.

## 2 . Чек-листы

Чек-лист - это список проверок. Это своего рода напоминание: «не забудь проверить то и то».

### **Пример чек-листов в жизни:**

- ✓ список покупок
- ✓ список вещей в отпуск

# Как оформить чек-лист?

Оформление может  
быть любым:

- список
- таблица
- схема
- .....

Имена		
Мужское		
Женское	✓	
Унисекс	✓	
Составное		
Пустое		

Проверка	Пример	Результат
Мужское	Иван	Успешная регистрация
Женское	Мария	Успешная регистрация
Унисекс	Саша	Успешная регистрация
Пустое		Ошибка: введите имя!

Это чек-лист

И это чек-лист

И даже это чек-лист



Если «Регистрация прошла успешно, на почту отправлено письмо-приветствие» - результат «ОК». (вынесли из таблицы общий результат, что бы не засорять таблицу)

Описание	Пример	Результат
Все поля заполнены правильно	Имя: Ольга Email: <b>ok.molechka@gmail.com</b> Пароль: 1	ОК
<b>Проверка поля «Имя»</b>		
Свое имя	Ольга	ОК
Короткое имя	Ева	ОК
Длинное имя (составное)	Розалинд Аруша Аркадина Алталун Флоренс Луна	ОК
...	...	...
<b>Проверка поля «Email»</b>		
Корректный email (популярный домен)	<b>olga@mail.ru</b>	ОК
Точка внутри email	<b>ok.molechka@gmail.com</b>	ОК
Кириллический email	<b>олечка@мустики.рф</b>	ОК
Пустая почта		Ошибка
Одно слово вместо домена	<b>olga@fdgfdg</b>	Ошибка
...		



## Особенности чек-листов

- минимум информации.
- независимые.

## Плюсы

- Меньше текста - проще и быстрее читать.
- Проще поддерживать - за счет предыдущих плюсов в основном. Если меньше текста, он медленнее устаревает.
- Быстрее писать - аналогично: меньше текста, больше дела!

## Минусы

- Не все поймут, как проводить тест.

## Чек-листы применимы, когда:

- вы - единственный тестирующий на проекте. Тогда нет смысла тратить время на подробное описание шагов, которые вы и так уже все наизусть знаете;
- система не суперсложная, и без указания конкретных шагов понятно, что и как делать.

## Есть еще ЧИТ-ЛИСТЫ ..... 😊

- Чек-лист - список проверок
- Чит-лист - список универсальных проверок

Подробнее о чит-листах

<https://habr.com/ru/articles/733986/>

<https://habr.com/ru/articles/715262/>

### **Чит-лист для проверки поля email**

- 1) Пустое поле email -> Сообщение о незаполненном поле email
- 2) Email в нижнем регистре -> Операция проводится успешно
- 3) Email в верхнем регистре -> Операция проводится успешно
- 4) Email с цифрами в имени аккаунта -> Операция проводится успешно
- 5) Email с цифрами в доменной части -> Операция проводится успешно
- 6) Email с дефисом в имени аккаунта -> Операция проводится успешно
- 7) Email с дефисом в доменной части -> Операция проводится успешно
- 8) Email со знаком подчеркивания в имени аккаунта -> Операция проводится успешно
- 9) Email со знаком подчеркивания в доменной части -> Операция проводится успешно
- 10) Email с точками в имени аккаунта -> Операция проводится успешно
- 11) Email с несколькими точками в доменной части -> Операция проводится успешно
- 12) Email без точек в доменной части -> Должно появиться сообщение о неправильном или некорректном e-mail введенном в поле
- 13) Превышение длины email (>320 символов) -> Должно появиться сообщение о неправильном или некорректном e-mail введенном в поле
- 14) Отсутствие @ в email -> Должно появиться сообщение о неправильном или некорректном e-mail введенном в поле
- 15) Email с пробелами в имени аккаунта -> Должно появиться сообщение о неправильном или некорректном e-mail введенном в поле.

ИТАК.....

## Тест-кейс

### Шаги

1. Открываем сайт <https://www.example.com/>
2. Нажимаем кнопку «Вход» в правом верхнем углу.
3. Устанавливаем курсор на поле «email».
4. Вводим значение «olga@mail.com».
5. Устанавливаем курсор на поле «Пароль».
6. Вводим значение «12345».
7. Нажимаем «Сохранить».

### Ожидаемый результат

Открылась главная страница сайта.

В верхнем правом углу отображается приветствие — «Здравствуйте, Ольга!»

## Чек-лист

- Авторизоваться через email

# ТЕСТ ДИЗАЙН (ТД)

---

Этап процесса тестирования ПО, на котором проектируются и создаются тестовые случаи (тест кейсы, чеклисты или тест сценарии), в соответствии с определёнными ранее критериями качества и целями тестирования.

Или проще:

Тест-дизайн - набор техник, которые позволяют приложить мало усилий и получить много результата.

**Цель техник тест дизайна** - Определить тестовые условия, тестовые случаи (проверки) и тестовые данные.

- **Тестовые условия:** Приложение корректно установлено и доступно для тестирования.
- **Тестовый случай:** проверить уникальность поля Login.
- **Тестовые данные:** в базе данных существует пользователь с Login=Ivan.

# Основные техники тест-дизайна

---

Английское название	Русский эквивалент
Equivalence Class Testing	Классы эквивалентности
Boundary Value Testing	Граничные значения
State-Transition Testing	Диаграмма состояний и переходов
Decision Table Testing	Таблица решений
User Case Testing	Варианты использования
Allpairs Algorithm testing	Попарное тестирование
Orthogonal Arrays Testing	Ортогональная матрица
...	...

# Классы эквивалентности

---

Два значения называются **эквивалентными**, когда приводят к одному результату. В таком случае неважно, взять значение А или Б для теста – так как они идентичны.

**Эквивалентный класс** — это одно или больше значений ввода, от которых ожидается сходное поведение, то есть они должны обрабатываться аналогичным образом. ..

А если реакция на все значения из одного класса эквивалентности одинаковая, то нет необходимости проверять все значения из этого класса.

Таким образом можно минимизировать количество тест-кейсов.

### Пример 1:

В поле процент прибыли пользователь вводит число. 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 ( $\leq 0$ )
- 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.**

Итого у нас есть 5 тест-кейсов для проверки: 50, -10, 146, aBc, !

## Пример 2:

Предположим, что книжный интернет магазин запускает новую кампанию "Больше тратишь — больше скидка".

Какие баги уже видны из этого условия ?

Потраченная сумма, руб-	Скидка, %
200 — 500	2
500—1000	3
1000 — 5000	4
5000 и более	5

**Баг1:** Непонятно, по какой ставке рассчитывается скидка, если потрачены следующие суммы: ровно 500 руб., ровно 1000 руб., ровно 5000 руб., так как каждая из этих сумм находится не в одной, а в двух корзинах со скидками.


**Баг 2:** Что означает "Потраченная сумма"? Это количество дензнаков, выплаченных только за книги, или полная сумма к оплате, включая оплату книг и расходы на доставку?

**Баг 3:** Для полноты картины нужно дописать эквивалентный класс от 0 до 199,99, на значения которого никакая скидка не распространяется.



Стоимость купленных книг, руб.	Скидка, %
0—199,99	0
200,00 — 499,99	2
500,00 — 999,99	3
1000,00 — 4999,99	4
5000,00 и более	5

Каждое значение внутри каждого класса является эквивалентным всем другим значениям этого класса. Потому что ко всем значениям класса должна применяться одинаковая логика кода.



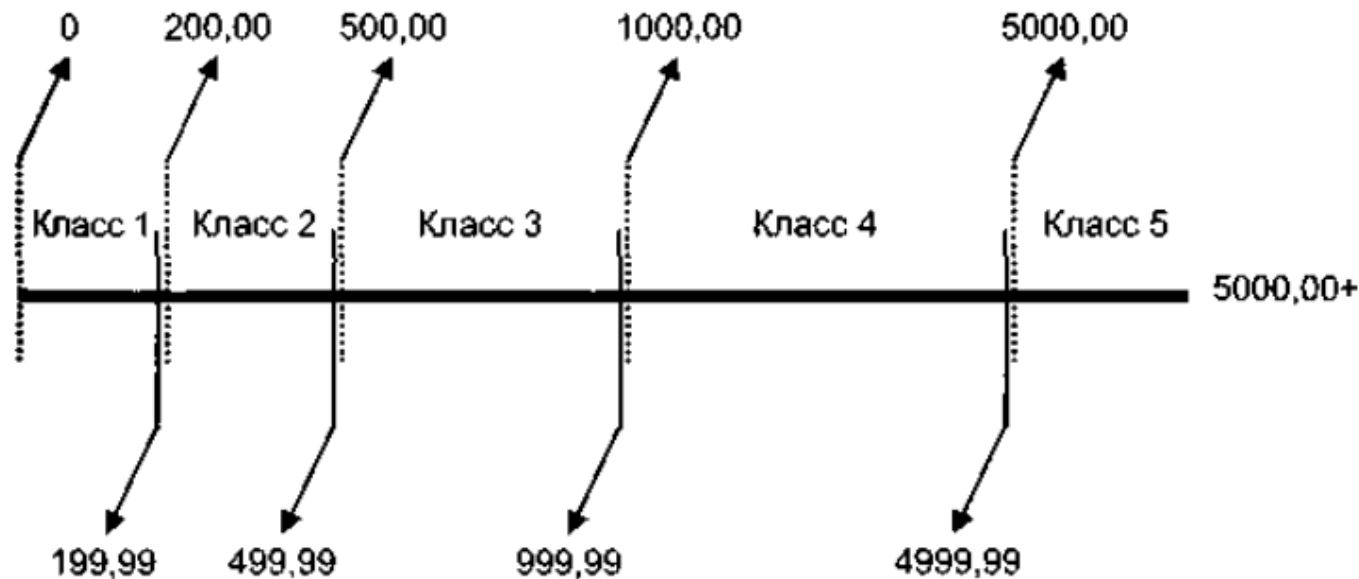
Класс 1:	0—199,99
Класс 2:	200,00 — 499,99
Класс 3:	500,00 — 999,99
Класс 4:	1000,00 — 4999,99
Класс 5:	5000,00 и более

Полезность раскладывания значений ввода на эквивалентные классы состоит в том, что мы отсеиваем огромное количество значений ввода, использовать которые для тестирования просто бессмысленно. Отсев происходит путем применения знаний о тестировании **граничных значений**.

# Граничные значения

Эта техника является "братом" разбиения на классы эквивалентности. Смысл этого подхода заключается в выборе значений на границах эквивалентных классов с минимальным шагом.

Вернемся к  
примеру  
книжного  
интернет-  
магазина



Поведение на границах эквивалентных областей имеет наибольшие шансы быть некорректным, таким образом **границы являются потенциальным источником дефектов.**

**Для каждого эквивалентного класса может быть лишь один из трех вариантов:**

- Есть только нижний предел (класс 5).
- Есть нижний и верхний пределы (класс 2, класс 3, класс 4).
- Есть только верхний предел (не рассматриваемый в данном примере класс, который ограничен только сверху отрицательным значением, непосредственно предшествующим классу 1).

# Тестирование граничных значений

1. Тестируется нижний предел данного класса (если он имеется).
2. Затем тестируется верхний предел данного класса (если он имеется).
3. Затем тестируется любое значение внутри данного класса.
4. Затем тестируется верхний предел класса, непосредственно предшествующего данному классу (если предшествующий класс имеется).
5. Затем тестируется нижний предел класса, непосредственно следующего за данным классом (если следующий класс имеется).

Можно  
записать  
граничные  
значения  
иначе

**Min -1**  
**Min**  
**Average**  
**Max**  
**Max +1**

Какие из этих тестов  
являются  
позитивными, а какие  
— негативными ?

Вернемся к примеру с книжным магазином

Протестируем эквивалентный класс 2.

Для покупок от 200,00 до 499,99 руб. (включительно) будет дана скидка 2%.

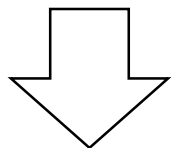
Если тестировать все значения из этого диапазона, то это 30 тыс вариантов (по количеству копеек в 299,99 руб. плюс один случай, когда потрачено 200,00 руб.)

По методике тестирования эквивалентного класса, нам нужно лишь

**5 вариантов данных:**

- |            |                    |
|------------|--------------------|
| 1. 200,00; | } Позитивные тесты |
| 2. 499,99; |                    |
| 3. 315,11; |                    |
| 4. 199,99; |                    |
| 5. 500,00. |                    |

Итак, 5 вариантов данных  
для одного класса  
эквивалентности.



1. 200,00;
2. 499,99;
3. 315,11;
4. 199,99;
5. 500,00.

А сколько будет  
вариантов данных  
для 5 классов ?

Класс 1:	0—199,99
Класс 2:	200,00 — 499,99
Класс 3:	500,00 — 999,99
Класс 4:	1000,00 — 4999,99
Класс 5:	5000,00 и более

Класс	Значение	Ожидаемая ставка скидки, %
Класс 1	0	0
	100,00	
	199,99	
Класс 2	200,00	2
	315,11	
	499,99	
Класс 3	500,00	3
	659,23	
	999,99	
Класс 4	1000,00	4
	3265,26	
	4999,99	
Класс 5	5000,00	5
	5075,00	

Для тестирования одного класса эквивалентности понадобилось 5 тестов, а если тестировать 5 классов понадобится **всего 14 тестов**, так как некоторые граничные условия будут учтены при тестировании других классов

## Пример: СИСТЕМА ПОДБОРА КРЕДИТА

Возраст до 18 лет



Кредит не разрешен

Возраст от 18 до 65 лет



- Потребительский кредит
- Кредит на недвижимость
- Кредит на покупку авто

Возраст старше 65 лет



Потребительский  
кредит

## Пример: СИСТЕМА ПОДБОРА КРЕДИТА

Класс эквивалентности #1: от 1 до 17

Возраст до 18 лет



Кредит не разрешен

Класс эквивалентности #2: от 18 до 65

Возраст от 18 до 65 лет



- Потребительский кредит
- Кредит на недвижимость
- Кредит на покупку авто

Класс эквивалентности #3: от 66 до 99

Возраст старше 65 лет



Потребительский  
кредит



## Пример: СИСТЕМА ПОДБОРА КРЕДИТА

Класс эквивалентности #1: от 1 до 17

Возраст до 18 лет



Кредит не разрешен

**Min -1:** пустое поле «Возраст»

**Min:** 1

**Average:** 11

**Max:** 17

**Max +1:** 18

> Сообщение об ошибке

> «Кредит не разрешен»

> «Кредит не разрешен»

> «Кредит не разрешен»

> «Потребительский кредит

Кредит на недвижимость

Кредит на покупку авто»

## Пример: СИСТЕМА ПОДБОРА КРЕДИТА

Класс эквивалентности #2: от 18 до 65 лет

Возраст до 18 до 65 лет 

- Потребительский кредит
- Кредит на недвижимость
- Кредит на покупку авто

**Min -1:** 17

**Min:** 18

**Average:** 40

**Max:** 65

**Max +1:** 66

> «Кредит не разрешен»

> 3 кредита

> 3 кредита

> 3 кредита

> «Потребительский кредит»

## Пример: СИСТЕМА ПОДБОРА КРЕДИТА

Класс эквивалентности #3: от 66 до 99 лет

Возраст старше 65 лет



Потребительский  
кредит

**Min -1:** 65  
**Min:** 66  
**Average:** 80  
**Max:** 99  
**Max +1:** 100

> 3 кредита  
> «Потребительский кредит»  
> «Потребительский кредит»  
> «Потребительский кредит»  
> Система не позволяет ввести  
больше двух символов

# Граничные значения для поля ДАТА РОЖДЕНИЯ

Формат: ДД.ММ.ГГГГ

День (значение):


	Январь, март, май, июль, август, октябрь, декабрь	Февраль (високосный год)	Февраль (не високосный год)	Апрель, июнь, сентябрь, ноябрь
Min – 1	00	00	00	00
Min	01	01	01	01
Average	15	15	15	15
Max	31	29	28	30
Max +1	32	30	29	31

День (формат):

Min -1: 1 символ    Min: 2 символа    Average: 2 символа    Max: 2 символа    Max+1: 3 символа

# E-MAIL

Формат: **name** **@** **hostname** **.** **domain**



	Local-part	@	Hostname	dot	domain
Min – 1	<i>Empty</i>	<i>Empty</i>	1 character	<i>Empty</i>	1 character
Min	1 character	@	2 characters	.	2 characters
Average	30 different characters	@	150 characters	.	6 characters
Max	64	@	243 characters	.	11 characters
Max +1	65	@ @	244 characters	..	12 characters

# E-MAIL

Формат: **local-part** @ **hostname.domain**  
Size: 256 characters

## ФОРМАТ

	Local-part	@	Hostname	dot	domain
Min – 1	<i>Empty</i>	<i>Empty</i>	1 character	<i>Empty</i>	1 character
Min	1 character	@	2 characters	.	2 characters
Average	30 characters	@	150 characters	.	6 characters
Max	64	@	243 characters	.	11 characters
Max +1	65	@ @	244 characters	..	12 characters

## РАЗМЕР

	Size	Examples
Min – 1	5 characters	@a.by; aa.by; a@.by; a@aby; a@a.b
Min	6 characters	a@a.by
Average	150 characters	...
Max	256 characters	...
Max +1	257 characters	...

# Тестирование ТАБЛИЦЫ РЕШЕНИЙ

---

- ✓ Таблица решений содержит все возможные комбинации для тестирования. Кроме того, она устраняет риски, обеспечивает качественное тестовое покрытие.
- ✓ **Каждый столбец (или строка) таблицы соотносится с бизнес-правилом, определяющим уникальную комбинацию условий и результат выполнения действий, связанных с этим правилом.**
- ✓ В тестировании таблица решений используется для того, чтобы на основе требований составить тест-кейсы. И ничего не забыть при сложных комбинациях входных условий! **Ведь каждая строка или столбец таблицы → готовый тест-кейс.**
- ✓ Стандартом покрытия для тестирования таблицы решений обычно является наличие хотя бы одного теста для каждой колонки (строки).

## Пример 1: таблица решений для выдачи кредита

Условия / Правила	Правило 1	Правило 2	Правило 3	Правило 4	
Возраст от 18 до 55 лет	да	нет	да	да	...
Ежемесячная выплата кредита меньше 1/3 ежемесячного дохода	да	да	нет	да	...
Человек трудоустроен	да	да	да	нет	...
Решения:	Кредит с обычной ставкой	Кредит не выдаем	Выдаем кредит с увеличенной на 1% ставкой	Кредит не выдаем	...
					...
					...



## Пример 2: составим таблицу для СИСТЕМЫ ПОКУПКИ ПРОЕЗДНЫХ БИЛЕТОВ

ВИД ТРАНСПОРТА:	АВТОБУС <input checked="" type="radio"/>	ТРОЛЛЕЙБУС <input type="radio"/>	МЕТРО <input type="radio"/>	ТРАМВАЙ <input type="radio"/>
ПЕРИОД:	ДЕНЬ <input type="radio"/>	НЕДЕЛЯ <input checked="" type="radio"/>	ДЕКАДА <input type="radio"/>	МЕСЯЦ <input type="radio"/>
СПОСОБ ОПЛАТЫ:		КАРТОЙ <input checked="" type="radio"/>	НАЛИЧНЫМИ <input type="radio"/>	
СУММА К ОПЛАТЕ:	10 РУБЛЕЙ			

ВИД ТРАНСПОРТА	ПЕРИОД	ОПЛАТА	ОЖИДАЕМЫЙ РЕЗУЛЬТАТ (СУММА)
Автобус	День	Карта	5 рублей
Автобус	Неделя	Карта	10 рублей
Автобус	Декада	Наличными	13 рублей
Автобус	Месяц	Наличными	50 рублей
Троллейбус	День	Наличными	5 рублей
Троллейбус	Неделя	Карта	10 рублей
Троллейбус	Декада	Наличными	13 рублей
Троллейбус	Месяц	Карта	50 рублей
Метро	День	Наличными	5 рублей
Метро	Неделя	Карта	10 рублей
Метро	Декада	Карта	13 рублей
Метро	Месяц	Наличными	50 рублей
Трамвай	День	Карта	5 рублей
Трамвай	Неделя	Наличными	10 рублей
Трамвай	Декада	Наличными	13 рублей
Трамвай	Месяц	Карта	50 рублей

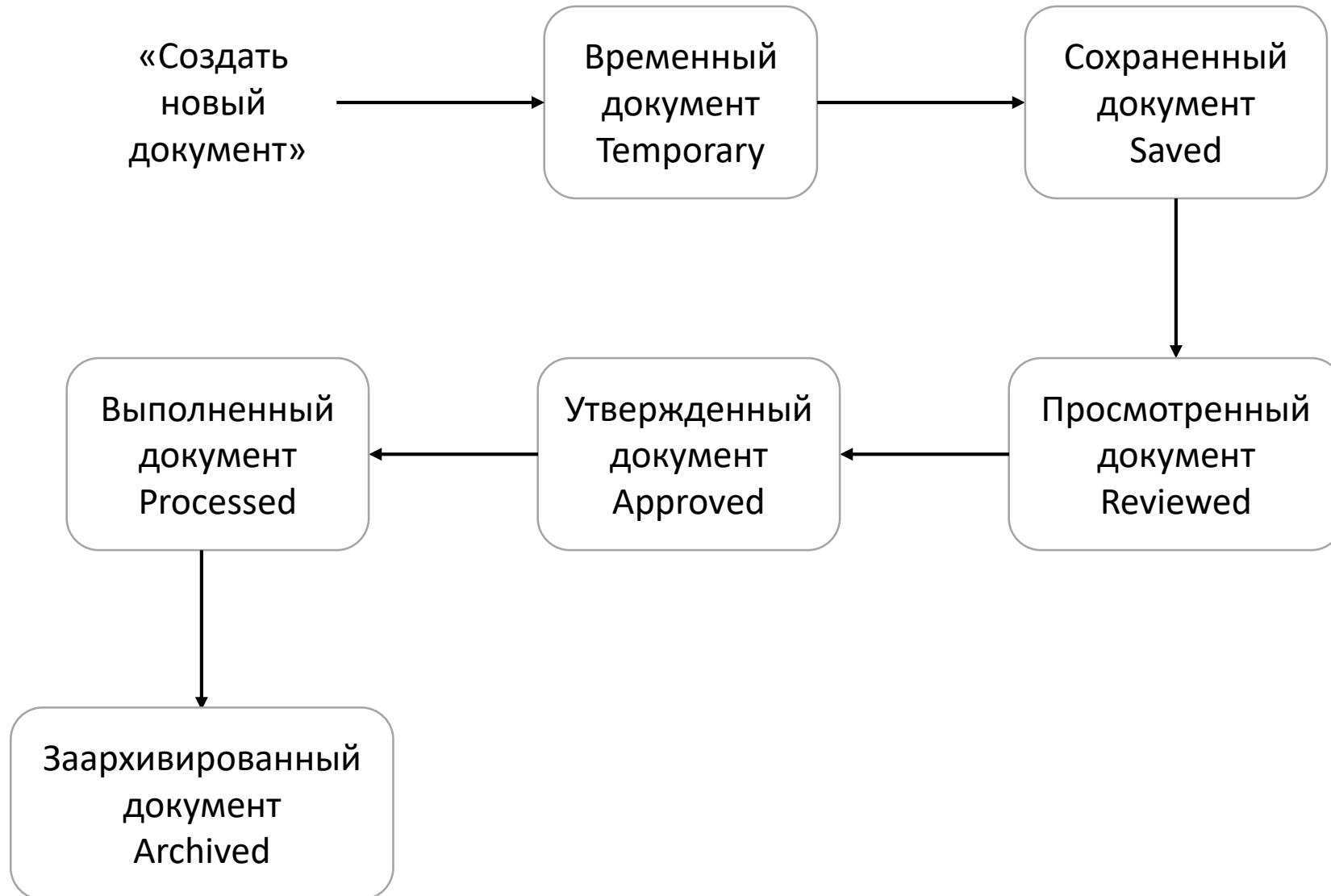
# Тестирование ТАБЛИЦЫ СОСТОЯНИЙ

---

Система может показывать различные отклики в зависимости от текущих условий или предшествовавшей истории состояний. Данный метод позволяет тестировщику рассматривать систему с точки зрения её состояний, переходов между состояниями, входов или событий, активизирующих изменения состояний (переходы) и действия, к которым приводят эти переходы. Состояния системы или тестируемого объекта разделяемы, определяемы и конечны.

Таблица состояний демонстрирует связи между состояниями и входами и может подсказать возможные некорректные переходы.

# СИСТЕМА ДОКУМЕНТООБОРОТА



Текущее состояние	Событие	Действие	Следующее состояние
null	«Создать новый документ»	Create	Temporary
null	«Создать новый документ»	Discard	null
Temporary	Заполнение документа	Save	Saved
Temporary	Заполнение документа	Cancel	null
Saved	Просмотр документа	Review	Reviewed
Saved	Просмотр документа	Reject	Rejected to creator
Reviewed	Утверждение документа	Approve	Approved
Reviewed	Утверждение документа	Reject	Archived
Approved	Выполнение действий по документу	Process	Processed
Processed	Архивация документа	Archive	Archived

# СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

Сценарий — это последовательная история, описывающая возможный способ использования приложения или системы.

Тестовый сценарий — это описание функциональности приложения или системы, которую нужно протестировать. Тестовые сценарии используются для сквозного тестирования функций и обычно разрабатываются на основе юзкейсов.

Тестовые сценарии служат основой для создания более детализированных тест-кейсов. Один тестовый сценарий может покрывать один или несколько тест-кейсов.



## Покупка товара

## Пример тестового сценария

1. Запустить приложение онлайн-магазина.
2. Проверить наличие поля поиска на главной странице.
3. Ввести название несуществующего товара в поле поиска.
4. Проверить, что система отображает сообщение об отсутствии результатов поиска.
5. Ввести название существующего товара в поле поиска.
6. Проверить, что система отображает результаты поиска с найденными товарами.
7. Выбрать один из найденных товаров.
8. Проверить, что система отображает подробную информацию о выбранном товаре.
9. Нажать кнопку «Добавить в корзину».
10. Проверить, что выбранный товар успешно добавлен в корзину.
11. Перейти в корзину, нажав на иконку корзины в верхнем меню.
12. Проверить, что система отображает содержимое корзины с выбранным товаром.
13. Нажать кнопку «Оформить заказ» в корзине.
14. Проверить, что система перенаправляет на страницу оформления заказа.
15. Заполнить форму с контактной информацией и адресом доставки.
16. Выбрать способ оплаты.
17. Нажать кнопку «Подтвердить заказ».
18. Проверить, что система обрабатывает заказ и отображает страницу подтверждения заказа с его уникальным номером.
19. Проверить, что информация о заказе на странице подтверждения соответствует ожидаемым данным.
20. Завершить покупку и выйти из приложения.

# ПОПАРНОЕ ТЕСТИРОВАНИЕ

---


**Попарное тестирование** это техника тест-дизайна методом черного ящика, при которой тест-кейсы создаются таким образом, чтобы выполнить **все возможные отдельные комбинации каждой пары входных параметров**.

Техника попарного тестирования очень помогает при разработке тестов для приложений, включающих множество параметров. Тесты разрабатываются таким образом, что для каждой пары входных параметров существуют все возможные комбинации этих параметров. Тестовые наборы (тест-сьюты, Test suite) охватывают все комбинации.

По статистике 87-93% ошибок возникает при встрече двух параметров. При большом сокращении количества тестов и времени на них можно пожертвовать 7 % допустимых ошибок



При попарном тестировании перебирают все возможные комбинации не между всеми параметрами, а между парами (на рисунке выделена каждая такая пара). Это позволяет значительно сократить количество тест-кейсов.



ФИО	Серия номер паспорта	Мобильный телефон	Дата рождения	Электронная почта	Чек-бокс отчества	Чек-бокс согласия
Валидное значение	Валидное значение	Валидное значение	Валидное значение	Валидное значение	Y	N
<u>Невалидное значение</u>	<u>Невалидное значение</u>	<u>Невалидное значение</u>	<u>Невалидное значение</u>	<u>Невалидное значение</u>	N	Y
Пусто значение	Пусто значение	Пусто значение	Пусто значение	<u>Невалидное значение</u>	Y	N
Валидное значение	<u>Невалидное значение</u>	Валидное значение	<u>Невалидное значение</u>	Валидное значение	N	Y
<u>Невалидное значение</u>	Пусто значение	<u>Невалидное значение</u>	Пусто значение	Валидное значение	Y	N
Пусто значение	Валидное значение	Пусто значение	Валидное значение	<u>Невалидное значение</u>	N	Y
Валидное значение	Пусто значение	Валидное значение	Пусто значение	Валидное значение	Y	N
<u>Невалидное значение</u>	Валидное значение	<u>Невалидное значение</u>	Валидное значение	<u>Невалидное значение</u>	N	Y
Пусто значение	<u>Невалидное значение</u>	Пусто значение	<u>Невалидное значение</u>	Валидное значение	Y	N

!!! Работа метода попарного тестирования подробно описана в статье по ссылке <https://habr.com/ru/companies/otus/articles/592575/>

Техника попарного тестирования помогает существенно уменьшить количество комбинаций проверок, **достаточных для обеспечения необходимого уровня качества программного обеспечения.**

Для генерации комбинаций тестов методом попарного тестирования можно использовать [онлайн-инструменты](#)

Однако эта техника имеет и некоторые ограничения. Она не работает, если:

- ✓ выбранные для тестирования значения некорректны;
- ✓ мало внимания уделяется комбинациям, которые могут привести к ошибке с высокой долей вероятности;
- ✓ взаимодействие между переменными недостаточно изучено.

# ТЕСТ ДИЗАЙН НА ОСНОВЕ ОПЫТА

---

Методы тестирования, основанные на опыте, знаниях и интуиции тестировщика.

- ✓ Предположения об ошибках;
- ✓ Тестирование, основанное на рисках;
- ✓ Исследовательское тестирование.