

Manual QA

# Техники тест-дизайна



# План занятия

- Классы эквивалентности
- Техника тест-дизайна анализ граничных значений
- Техника тест-дизайна попарного тестирования
- Техника тест-дизайна таблиц принятия решений



# ОСНОВНОЙ БЛОК





# Классы ЭКВИВАЛЕНТНОСТИ



# Экспресс-опрос



# Экспресс-опрос

?

Как вы поняли, почему невозможно всеобъемлющее тестирование?



## Техники тест-дизайна

Это подходы, которые позволяют ограничить набор реализуемых проверок, в значительной степени не повышая риск появления багов в непроверяемых случаях.

# Экспресс-опрос





# Экспресс-опрос



Select an Age combobox - обязательное для заполнения поле, позволяющая выбрать из списка значения от 1 до 100. В зависимости от значения программа выдает результат: "Совершеннолетний" либо "Несовершеннолетний".

Подскажите варианты вводимых данных для тестирования.

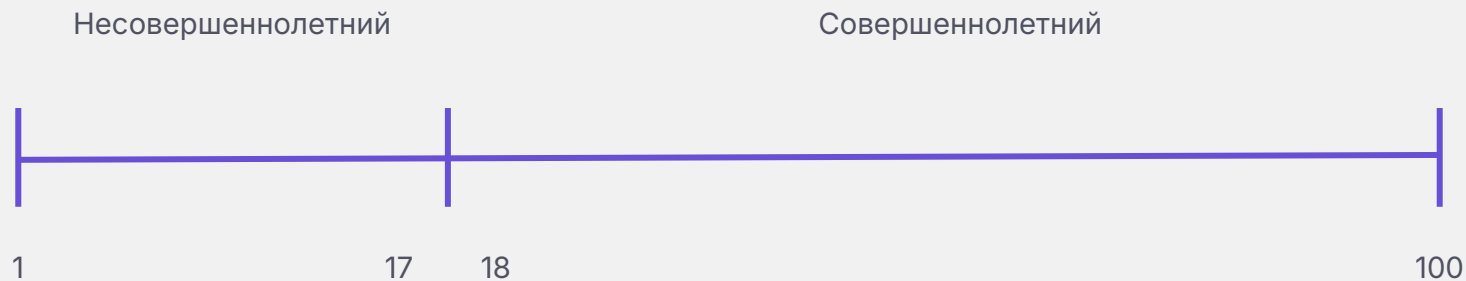
The image shows a screenshot of a web form. It features a label "Select an Age:" followed by a combobox. The combobox has a light gray border and a small downward arrow on the right side. The value "12" is displayed inside the combobox, indicating it is the selected option.



## Классы эквивалентности

Набор данных, который запускает одни и те же модули и должен приводить к одним и тем же результатам.

# Классы эквивалентности



# Пример

Представим, что тестируем модуль для HR, который определяет брать на работу кандидата или нет, базирясь на возрасте кандидата.







Условия:

- от 0 до 16: не нанимать
- от 16 до 18: можем нанять только на частичную занятость
- от 18 до 55: можем нанять на полную занятость
- от 55 до 99: не нанимать

# Классы эквивалентности валидных данных

- 1 от 0 до 16
- 2 от 17 до 18
- 3 от 19 до 55
- 4 от 56 до 99

# Классы эквивалентности невалидных данных

-  Отрицательные значения
-  Значения более 99
-  Дробные значения
-  Символьные значения
-  Пустой ввод
-  ...

# Валидные классы нескольких переменных объединяются в один тест-кейс



## Почему?

- Все валидные значения предполагают, что приложение работает корректно в любых таких комбинациях.
- Проверка одной комбинации валидных данных достаточно, чтобы убедиться в базовой функциональности.

## Пример

Есть две переменные:

- **Возраст (Age):** валидные классы – от 1 до 100.
- **Имя (Name):** валидные классы – любое непустое значение.

Один тест-кейс для проверки валидных классов может выглядеть так:

- **Возраст = 25** (из валидного диапазона возраста).
- **Имя = "Иван"** (валидное непустое имя).

# Невалидные классы тестируются отдельно



## Почему?

- Невалидные данные могут вызывать различные ошибки, поэтому каждый такой случай должен быть обработан системой независимо.
- Разделение невалидных тестов помогает локализовать проблему.

## Пример

- **Возраст (Age):** невалидные классы – меньше 1 или больше 100 (например, -5, 150).
- **Имя (Name):** невалидные классы – пустое значение или слишком длинное имя (например, "").

Отдельные тест-кейсы:

1. **Возраст = -5, имя = "Иван"** (проверка невалидного возраста).
2. **Возраст = 25, имя = ""** (проверка невалидного имени).



# Валидные и невалидные классы

## Валидные классы

Объединяются в один тест-кейс, потому что система должна обрабатывать любые валидные комбинации одинаково

## Невалидные классы

Тестируются отдельно, чтобы проверить, как система обрабатывает каждую ошибочную ситуацию



# ВОПРОСЫ





# Техника тест- дизайна анализ граничных значений



## Граничные значения

Это значения, в которых один класс эквивалентности переходит в другой.



## Three-point approach

Это метод тестирования, при котором проверяются три значения для каждой границы диапазона. Такой подход позволяет выявить ошибки, которые могут возникнуть при обработке граничных случаев.

# Three-point approach

Граничное значение

Это значение точно на границе  
диапазона

Проверяет, правильно ли система  
распознает значение как допустимое  
или недопустимое

Значение перед границей

Значение после границы

# Three-point approach

Граничное значение

Значение перед границей

Значение после границы

Это значение, которое меньше (для нижней границы) или больше (для верхней границы) на минимальное возможное изменение

Проверяет, правильно ли система обрабатывает значения, выходящие за пределы диапазона

# Three-point approach

Граничное значение

Значение перед границей

Значение после границы

Это значение, которое больше (для нижней границы) или меньше (для верхней границы) на минимальное возможное изменение

Проверяет, правильно ли система обрабатывает значения, находящиеся в пределах диапазона



# Пример

Есть диапазон допустимых значений возраста от **18** до **65**.

## Для нижней границы (18):

- **Значение перед границей:** 17 (недопустимое значение).
- **Граничное значение:** 18 (допустимое значение).
- **Значение после границы:** 19 (допустимое значение).

## Для верхней границы (65):

- **Значение перед границей:** 64 (допустимое значение).
- **Граничное значение:** 65 (допустимое значение).
- **Значение после границы:** 66 (недопустимое значение).

# Важность Three-point approach



Покрытие крайних случаев



Выявление логических ошибок



Экономия времени



## Two-point approach

Это метод тестирования, при котором для каждой границы диапазона проверяются два значения:

- Граничное значение.
- Ближайшее значение за границей.

# Two-point approach

Для нижней границы

Граничное значение: минимальное значение,  
допустимое в диапазоне

Значение за границей: значение сразу ниже  
границы, которое выходит за пределы диапазона

Для верхней границы

Граничное значение: максимальное значение,  
допустимое в диапазоне

Значение за границей: значение сразу выше  
границы, которое выходит за пределы диапазона

# Пример

Диапазон допустимых значений: от **10** до **20**.

## Нижняя граница (10):

- Граничное значение: 10 (должно быть обработано как допустимое).
- Значение за границей: 9 (должно быть отклонено как недопустимое).

## Верхняя граница (20):

- Граничное значение: 20 (должно быть обработано как допустимое).
- Значение за границей: 21 (должно быть отклонено как недопустимое).

# Задача: HR-модуль

## Условия:

от 0 до 16: не нанимать

от 16 до 18: можем нанять только на частичную занятость

от 18 до 55: можем нанять на полную занятость

от 55 до 99: не нанимать

## Наборы проверок:

- {-1, 0, 1}
- {15, 16, 17}
- {17, 18, 19}
- {54, 55, 56}
- Негативные {-42, 1001, HELLO, %\$#@}

# Экспресс-опрос



# Экспресс-опрос



В зависимости от суммы на текущем счету в банке, ежемесячно рассчитывается бонус:

- От \$100.00 до \$1000.00 (включительно) : +3%
- От \$1000.00 до \$10 000.00 : +5%
- \$10 000.00 и больше : +7%

Назовите значения для проверки граничных условий для суммы в \$1000?



# Экспресс-опрос



В зависимости от суммы на текущем счету в банке, ежемесячно рассчитывается бонус:

- От \$100.00 до \$1000.00 (включительно) : +3%
- От \$1000.00 до \$10 000.00 : +5%
- \$10 000.00 и больше : +7%

Назовите значения для проверки граничных условий для суммы в \$1000?





# ВОПРОСЫ





# ЗАДАНИЕ



**Upload a new file \***

Choose File

No file chosen

Upload

Files must be less than 2 MB.

Allowed file types: png gif jpg jpeg.

**Разделить** пространство значений переменной **на группы**:

Upload:

разные размеры файла

разные имена файлов

разное содержимое файлов

разное количество файлов

**Upload a new file \***

Choose File

No file chosen

Upload

Files must be less than 2 MB.

Allowed file types: png gif jpg jpeg.

## SEND FEEDBACK

Info:

We would like your feedback to improve our service!

Data:

Write your feedback, up to 1000 symbols:

Send



# Техника тест-дизайна попарного тестирования



## Попарное тестирование

Это техника оптимизации тестового покрытия, используемая для проверки взаимодействия между парами значений нескольких параметров системы. Этот метод помогает значительно уменьшить количество тест-кейсов, сохраняя при этом высокую вероятность нахождения дефектов.



# Задача: приложение для заказа автомобиля

- С помощью приложения можно покупать и продавать машины. Приложение должно поддерживать оказание услуг в **Берлине** и **Мюнхене**.
- В приложении должны содержаться регистрационные номера, которые могут быть **валидными** и **невалидным**.
- Оно должно разрешать продажу следующих марок автомобилей: **BMW**, **Audi** и **Mercedes**.
- Доступны два типа бронирования: бронирование через **интернет** и в **офлайн-магазине**.
- Заказы доступны к размещению **только** в рабочие часы.

# Варианты для вычисления количества существующих комбинаций

- |   |  |
|---|--|
| <ol style="list-style-type: none"> <li>1. Категория заказа               <ul style="list-style-type: none"> <li>• Купить</li> <li>• Продать</li> </ul> </li> <li>2. Местоположение               <ul style="list-style-type: none"> <li>• Берлин</li> <li>• Мюнхен</li> </ul> </li> <li>3. Марка автомобиля               <ul style="list-style-type: none"> <li>• BMW</li> <li>• Audi</li> <li>• Mercedes</li> </ul> </li> </ol> | <ol style="list-style-type: none"> <li>4. Регистрационные номера               <ul style="list-style-type: none"> <li>• Валидные (5000)</li> <li>• Невалидные</li> </ul> </li> <li>5. Тип заказа               <ul style="list-style-type: none"> <li>• Заказ через Интернет</li> <li>• Заказ в магазине</li> </ul> </li> <li>6. Время заказа               <ul style="list-style-type: none"> <li>• Рабочие часы</li> <li>• Нерабочие часы</li> </ul> </li> </ol> |
|---|--|

**$2 \times 2 \times 3 \times 5001 \times 2 \times 2 = 240\,048$  комбинаций**

# Использование техники разбиения на классы эквивалентности

Сокращаем регистрационные номера до двух типов:

1. Валидный регистрационный номер
2. Невайдный регистрационный номер

**Было:  $2 \times 2 \times 3 \times 5001 \times 2 \times 2 = 240\ 048$**

**Стало:  $2 \times 2 \times 3 \times 2 \times 2 \times 2 = 96$**



## All Combinations Testing (исчерпывающее тестирование)

Техника тест-дизайна, при которой ты  
**проверяешь все возможные комбинации  
значений всех параметров.**

**$2 \times 2 \times 3 \times 5001 \times 2 \times 2 = 240\,048$  проверок**



## All Combinations Testing + классы ЭКВИВАЛЕНТНОСТИ

$2 \times 2 \times 3 \times 2 \times 2$  (вместо 5001)  $\times 2 = 96$  проверок



## Pairwise

**$2+2+3+2+2+2 = 13$  проверок**



## Pairwise

**$2+2+3+2+2+2 = 13$  проверок**

№	Категория	Местоположение	Марка	Рег. номер	Тип бронирования	Время заказа
1	Купить	Берлин	BMW	Валидный	Интернет	Рабочие часы
2	Продать	Мюнхен	Audi	Невалидный	В магазине	Рабочие часы
3	Купить	Мюнхен	Mercedes	Валидный	В магазине	Нерабочие часы
4	Продать	Берлин	BMW	Невалидный	Интернет	Нерабочие часы
5	Купить	Берлин	Audi	Валидный	В магазине	Нерабочие часы
6	Продать	Мюнхен	Mercedes	Невалидный	Интернет	Рабочие часы
7	Купить	Мюнхен	BMW	Валидный	Интернет	Нерабочие часы
8	Продать	Берлин	Audi	Невалидный	В магазине	Рабочие часы
9	Купить	Берлин	Mercedes	Валидный	В магазине	Рабочие часы
10	Продать	Мюнхен	BMW	Невалидный	Интернет	Нерабочие часы
11	Купить	Берлин	Mercedes	Невалидный	В магазине	Рабочие часы
12	Продать	Мюнхен	Audi	Валидный	В магазине	Нерабочие часы
13	Купить	Мюнхен	BMW	Невалидный	Интернет	Рабочие часы

ICN  
IT CAREER HUB





**Each Choice** – метод тест-дизайна, при котором для каждого параметра (или фильтра) тестируются **все возможные значения** поочередно, независимо от того, как эти значения комбинируются с другими параметрами. То есть, для каждого фильтра выбираются все его возможные опции, и проверяется каждый вариант.

**3 проверки**

# Задача: приложение для заказа автомобиля

Категория заказа	Местоположение	Марка авто	Регистрационный номер	Тип заказа	Время заказа
Покупка	Берлин	BMW	Валидные	Онлайн	Рабочие часы
Продажа	Мюнхен	Audi	Невалидные	Офлайн	Нерабочие часы
		Mercedes			



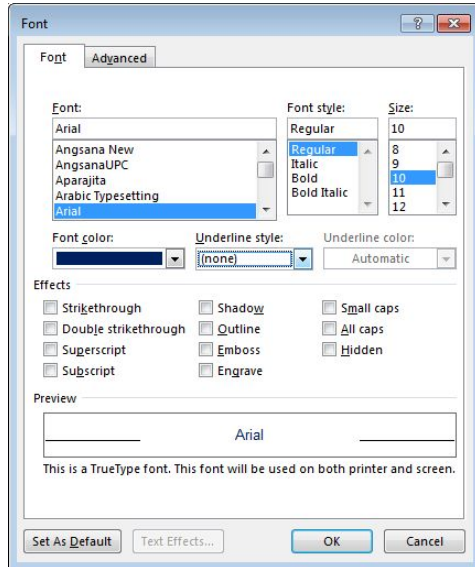
# ВОПРОСЫ



# Экспресс-опрос



# Экспресс-опрос



## Задача

Сосчитайте количество всех проверок техникой Pairwise и Each Choice



# Техника тест- дизайна таблиц принятия решений



## Таблица принятия решений (Decision Table Testing)

Это техника тестирования, которая используется для анализа сложных бизнес-логик, где необходимо учитывать комбинации входных условий и действий. Она помогает систематизировать тестирование, покрывая все возможные комбинации условий и связанных с ними результатов.

# Пример структуры таблицы принятия решений



## Условия

1. Клиент зарегистрирован?
2. У клиента есть скидка?
3. Сумма заказа превышает 100 долларов?

## Действия

1. Предоставить скидку.
2. Применить бесплатную доставку.
3. Отклонить заказ.



# Таблица принятия решений

Условие 1	Условие 2	Условие 3	Действие 1	Действие 2	Действие 3
Нет	Нет	Нет	Нет	Нет	Да
Да	Нет	Да	Да	Да	Нет
Да	Да	Нет	Да	Нет	Нет
Да	Да	Да	Да	Да	Нет

# Задача: система управления заказами

Вы тестируете систему управления заказами, где:

- Если клиент зарегистрирован и сумма заказа превышает 100 долларов, предоставляется скидка.
- Если сумма заказа превышает 200 долларов, предоставляется бесплатная доставка.
- Если клиент не зарегистрирован, заказ отклоняется.

# Задача: система управления заказами

Клиент зарегистрирован	Сумма > 100	Сумма > 200	Скидка	Бесплатная доставка	Отклонить заказ
Нет	Нет	Нет	Нет	Нет	Да
Нет	Да	Нет	Нет	Нет	Да
Да	Нет	Нет	Нет	Нет	Нет
Да	Да	Нет	Да	Нет	Нет
Да	Да	Да	Да	Да	Нет

# Задача: система управления заказами

Когда применять:

- Когда система имеет сложные бизнес-правила с множеством зависимостей.
- Когда нужно проверить, что покрыты все возможные сценарии для набора условий.
- Когда логика системы должна быть понятной для всех участников команды (включая бизнес-аналитиков).

# Шаблон таблиц для работы над задачами

Шаблон





# ВОПРОСЫ



# Домашнее задание

Составить таблицу принятия решений для анализа всех возможных случаев для следующих требований:

1. Политика скидок компании зависит от того, является ли клиент новым или постоянным.
2. Скидка предоставляется пенсионерам. Новые клиенты получают скидку 20% .
3. На второй заказ, а постоянные клиенты получают бесплатную доставку.
4. Клиенты старше 65 лет (пенсионер) также получают скидку 10% на текущую покупку.

[Шаблон для решения](#) во вкладке Discount.



## Error Guessing

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.







## State-Transition

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.



# ЗАДАНИЕ

