Также читай здесь - <https://docs.google.com/document/d/19MJqSZTet7g5WQ6U_tuo2X08kRCVR7GQ4D2y9KrxFdg/edit?usp=sharing>

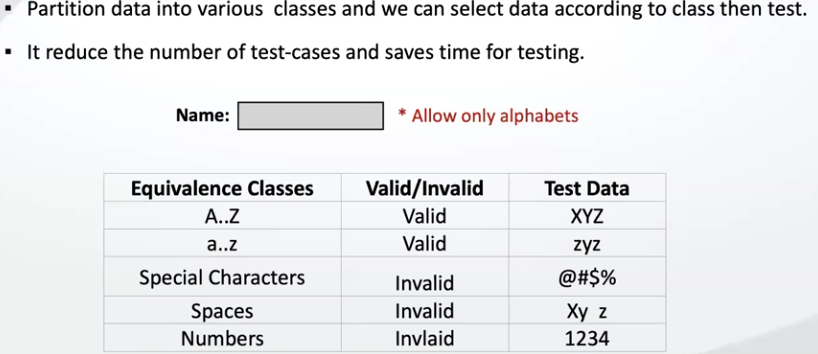
Test design techniques helps to design better cases. It reduce the number of test cases to be executed. Хорошего тестировщика от плохого отличает умение проектировать оптимальные тесты, которые за минимальное количество шагов обеспечат максимальное тестовое покрытие. Именно для этого используется тест-анализ и тест-дизайн.

Зачем тестировать все значения из одного диапазона, если они выдают один и тот же результат?

Зачем тестировать все параметры, если достаточно проверить только взаимодействие пары из них?

Зачем писать много текста для проверки двух правил, если можно создать одну таблицу и поместить в нее результаты?

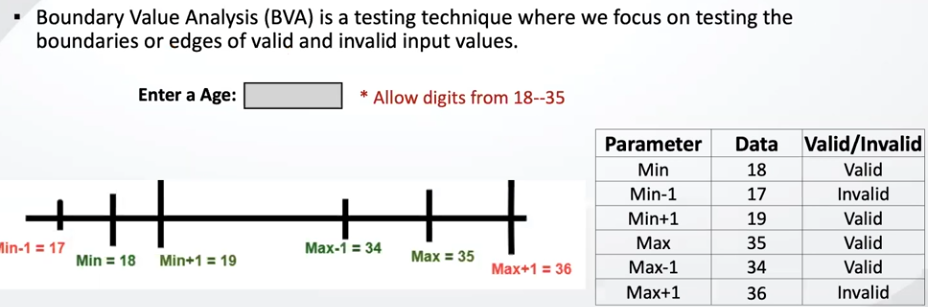
**1 Equivalence Class Partitoning**

****

This technique is applicable for input boxes or text boxes more often. Accordingly to requirements all the test data is divided into various classes or various categories and each category we need to specify as valid or invalid and accordingly we will prepare the test. When we pass a valid date it should accept, when we pass invalid data it should not accept, this is the correct behavior.

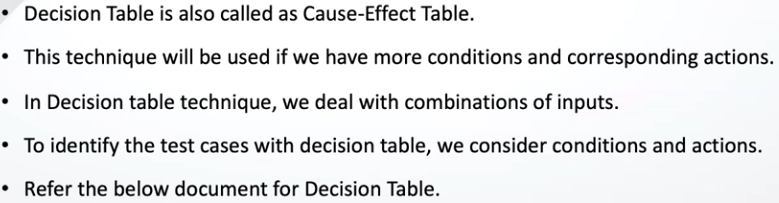
In that example, as you can see, presented positive and negative input it is important because only positive input is not enough.

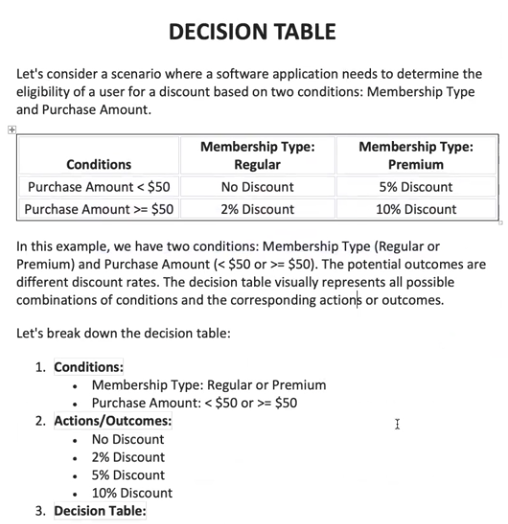
**2 Boundary Value Analysis (BVA) (пограничные значения)**

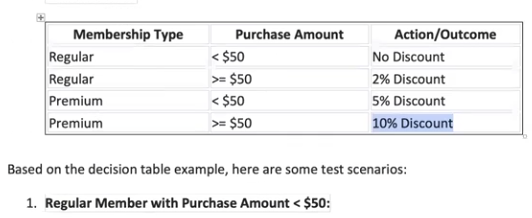
****

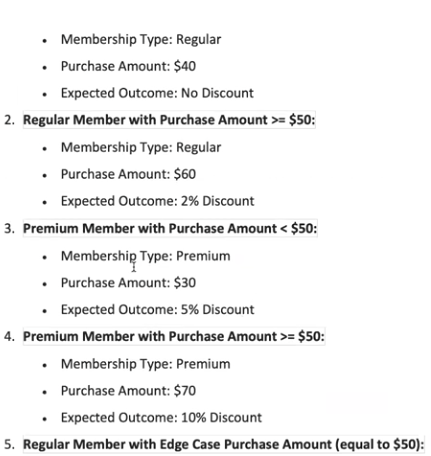
В примере выше поле принимает числа от 18 до 35 включительно, соответственно нам надо проверить что ниже 18 числа не будут приниматься, также не должны приниматься числа выше 35. Но делать тесты с 0 по 17 это долго, дорого и трудозатратно, неговоря чтобы тестировать все числа после 35. Для этого и существует эта техника, она берёт пограничные величины, если величина 18 то надо взять 18, потом 18-1 и 18+1, тоже самое сделать и с верхним пограничным числом.

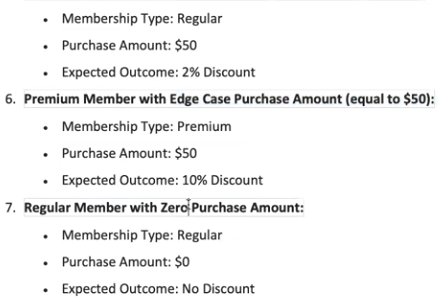
**3 Decision Table based testing**

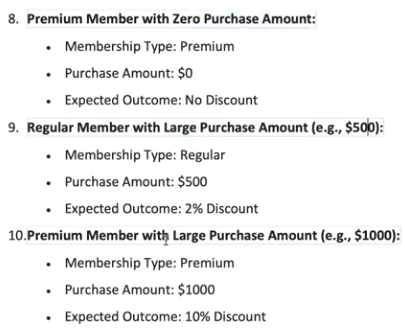






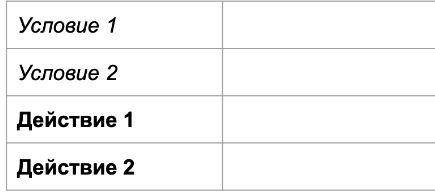






Ещё пример:

Пары, условий и действий образуют строки таблицы, при этом условия указываются сверху, а действия – снизу. действия -это ожидаемый результат в этой технике



Оклад сотрудника рассчитывается следующим образом:

без опыта и высшего образования - 500

с высшим образованием, но без опыта - 600

без высшего образования, с опытом - 600

с опытом и высшим образованием - 700

Премия для сотрудника рассчитывается следующим образом:

без опыта и высшего образования - 5%

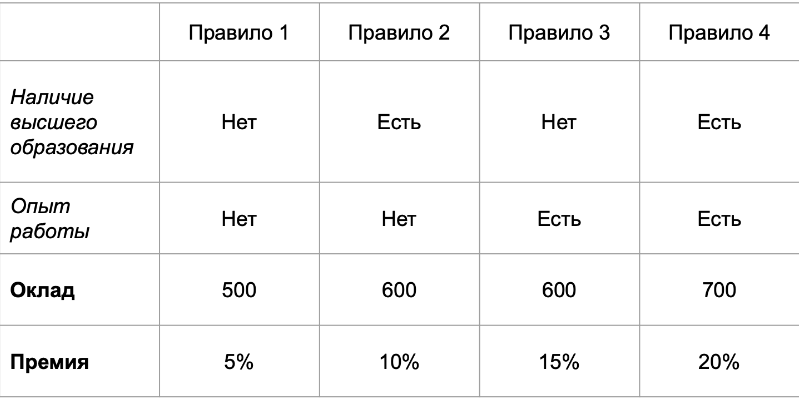
с высшим образованием, но без опыта - 10%

без высшего образования, с опытом - 15%

с опытом и высшим образованием - 20%

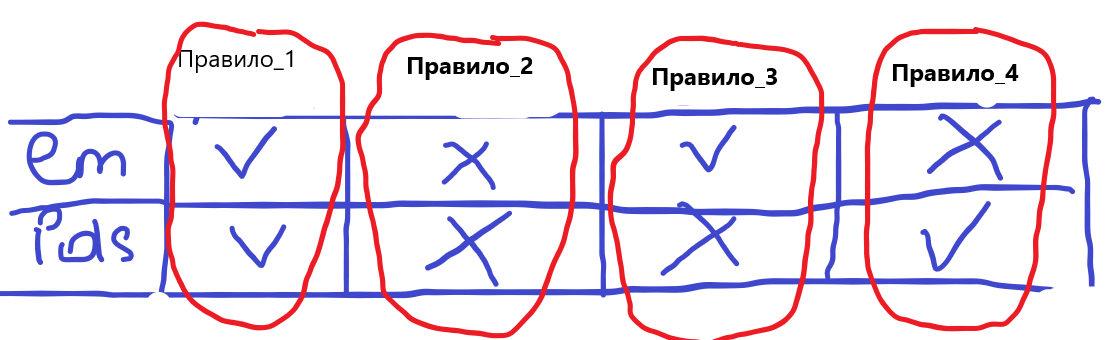
Ваша задача протестировать приложение, которое должно рассчитывать оклад и премию сотрудника.

Так как в данном случае можно четко определить условия, действия и правила, то это замечательный кандидат для создания таблицы принятия решений.



Ещё пример:

Есть два поля: email и пароль. И два значения для каждого поля: валидное и невалидное. Какое количество проверок необходимо провести в рамках таблицы принятия решений?



Получается 4 возможных сетов проверок -

(да/да),

(нет/нет),

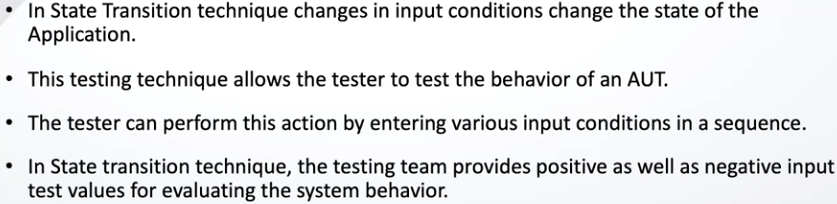
(да/нет),

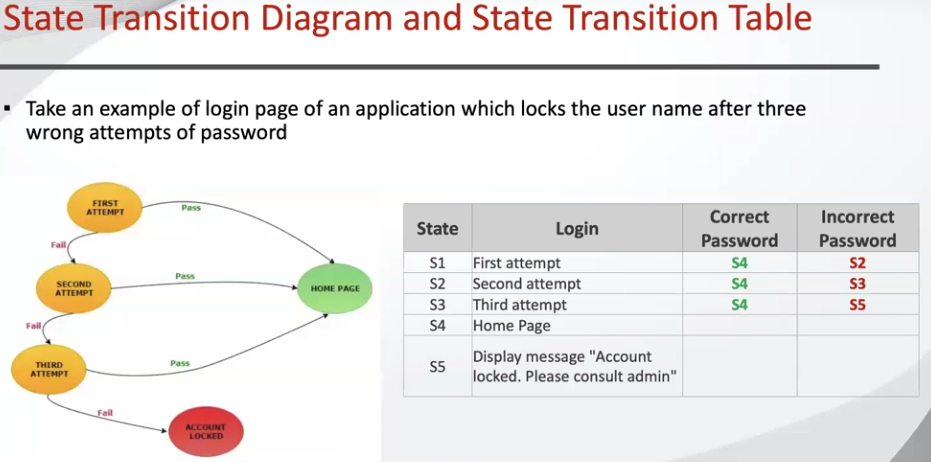
(нет/да)

Здесь на этом рисунке не указаны действия (действия -это ожидаемый результат в этой технике)

**4 State transition**

This technique will be used when the conditions that are keep changing will change the state of the AUT





In that technique based on the requirements first we need to draw the diagram, in the example above we have a login page. the first time when a user enters a wrong password and login it will redirect him to the initial step, for the second and the third time it will be the same, but at 4-th time it will block the user. These are requirements. Based on the action of the user the condition of the application will change, now wen we know the requirements we need to draw the diagram. See the picture above. Now if you look at the diagram we can easily recognise the flow, the actions that we need to perform, now we can develop the State Transition Table.

Таблица читается по строкам. s1 это первая попытка, если введён пассворд корректно то приложение перекидывает тебя на s4 (home page), если пасворд некорректный то на s2 (second attempt).

Как тестировщики нам дадут только требования, всё остальное надо рисовать самому.

**Попарное тестирование (pairwise testing)** - техника тестирования методом черного ящика, при которой тестовые сценарии разрабатываются таким образом, чтобы выполнить тестирование для всех комбинаций пар параметр-значение. Для формирования пар используется два подхода: ортогональные массивы и алгоритм All Pairs. В практике тестирования вы будете использовать последний, поэтому давайте сконцентрируемся на нем.

All Pairs

Алгоритм All Pairs позволяет генерировать все пары автоматически. Самым популярным инструментом для этого является PICT, но существует еще несколько десятков других решений.

Выходные данные полученные этим способом могут отличаться от данных, полученных в результате применения ортогональных массивов. Про ортогональные массивы можно прочитать в пособии Lee Copeland, «A practitioner’s guide to software test design».

Пример использования

Давайте рассмотрим пример применения этого алгоритма с тестированием фильтрации на примере интернет-магазина.

Пользователь может осуществить фильтрацию по следующим параметрам:

Цвет: красный, белый, черный, серый

Размер: S, M, L, XL

Пол: мужской, женский, унисекс

Доставка: есть, нет

Общее количество всех проверок без применения попарного тестирования: 4\*4\*3\*2 = 96.

Согласитесь, что проводить 96 тестов всех комбинаций времязатратно. Для оптимизации нам пригодится Pairwise.

Согласно этой технике, необходимо протестировать взаимодействие всех пар параметров и их значений, что будет соответствовать практически полному тестовому покрытию, если бы мы тестировали все комбинации.

Какие шаги нужно сделать?

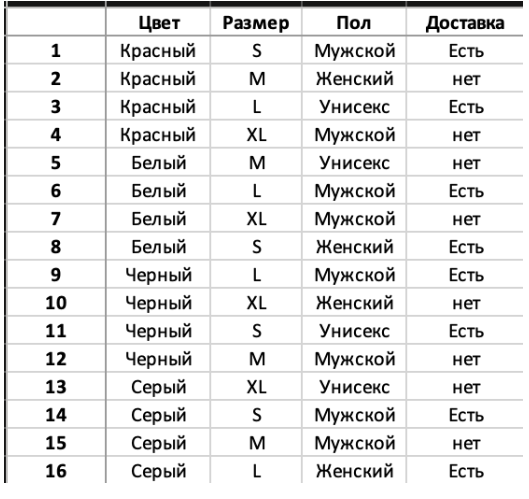
Для выполнения задания будем использовать ресурс https://pairwise.teremokgames.com, который не требует предварительной установки.

1. Создадим таблицу с параметрами и значениями. Всегда начинаем с параметра, у которого больше всего значений.



2. Для генерации пар необходимо нажать кнопку

3. После нажатия формируется таблица с необходимыми тестовыми данными. Количество проверок сократится с 96 до 16. Таблицу необходимо использовать для создания чек-листа или тест-кейсов



5 Error Guessing