Software Engineering Process and Quality Management

Specification-Based Test Case Design Techniques

Learning Outcomes

Specification based testing techniques

- Equivalence partitioning

- Boundary value analysis

- Decision table

Specification Based Testing Technique

- Discovering what claims are made in the specifications and testing the product against them.

- Only concentrates on what the software does, not how it does it.

- Do not require the knowledge of how the system or component is structured.

- Also known as Behavior Based Testing and Black Box Testing.

Specification Based Testing Design Technique

- Uses the specification of the program as the point of reference for test data selection and adequacy.

- A specification can be anything like a written document, collection of use cases, a set of models, or a prototype.

Types of Specification Based Testing Techniques

- Equivalence Partitioning

- Boundary Value Analysis

- Decision Tables

- State Transitioning

Equivalence Partitioning

- A technique to ensure the maximum test coverage using minimal test cases .

- Divides the input data of a software unit into partitions of equivalent data from which test cases can be derived.

- The idea behind the technique is to divide a set of test conditions into groups or sets that can be considered as same.

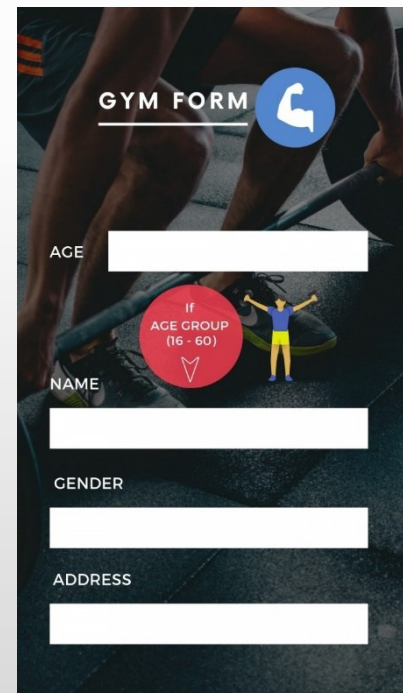# Equivalence Partitioning (Cont.)

- Equivalence partitioning works on certain assumptions.

    1. The system will handle all the test input variations within a partition in the same way.

    2. If one of the input condition passes, then all other input conditions within the partition will pass as well.

    3. If one of the input conditions fails, then all other input conditions within the partition will fail as well.

- The success and effectiveness of Equivalence partitioning lie in how reasonable the above assumptions hold.

Example

Fitness1st gym provides membership for the clients between the age of 16 to 60. It has the following online application form for getting the gym membership.

In this form;

- User has to fill the age first.
- User can go further only if the age is between 16 to 60.
- Otherwise, there will be a message saying that you cannot get a membership.

Testing the form

- Form has to be tested with values;
  - Less than 16
  - Between 16-60
  - More than 60

- How many combinations are required to say that the functionality works safely?

  - <16 has 15 combinations from 0-15. More conditions can be added by considering the negative values.

  - 16-60 has 45 combinations

  - >60 has 40 combinations (if only consider till 100)

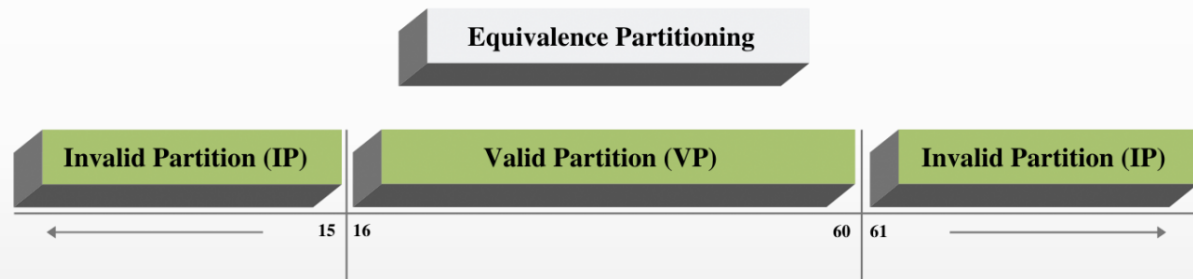- Is it possible to test all these 100+ combinations?

How to do Equivalence Partitioning? (Cont.)

Divide (partition) the input values into sets of valid and invalid partitions.
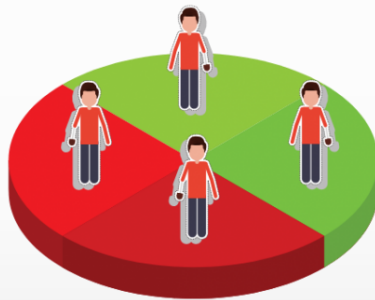
Equivalence Partitioning

- Valid partitions are values that should be accepted by the component or system under test. This partition is called "**Valid Equivalence Partition**."

- Invalid partitions are values that should be rejected by the component or system under test. This partition is called "**Invalid Equivalence Partition**."

How to do Equivalence Partitioning? (Cont.)

- Test conditions required to test the system:

  ✓ Enter Age = 5

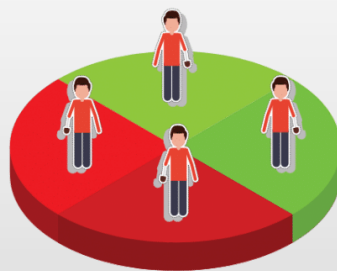 ✓ Enter Age = 20

  ✓ Enter Age = 65
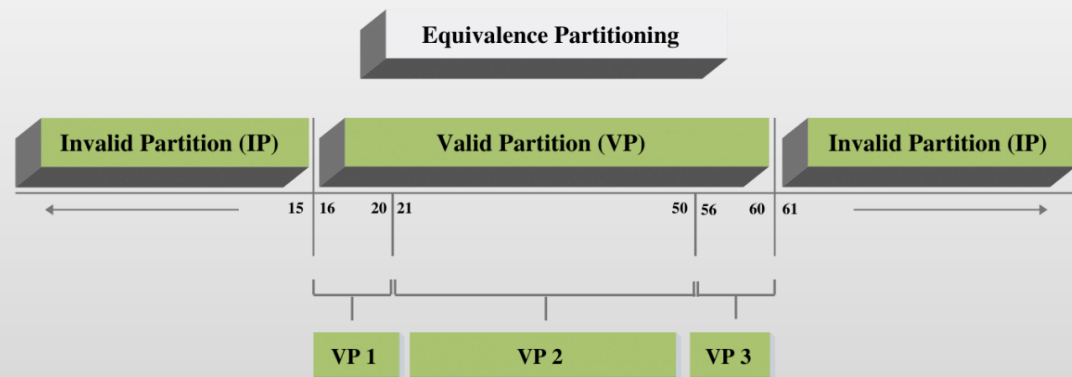
How to do Equivalence Partitioning? (Cont.)

- These partitions can be further divided into sub-partitions if required.

Assume that if the age is 16-20 years old or 55-60 years old, there is an additional requirement to attach a proof of the age, while submitting the membership form.

In this case, partition will look like below;

Equivalence Partitioning

How to do Equivalence Partitioning? (Cont.)

- VP1, VP2, and VP3 are all valid sub partitions based on the additional requirements.

- Therefore, test conditions could be:
  - ✓ Enter Age = 5
  - ✓ Enter Age = 18
  - ✓ Enter Age = 30
  - ✓ Enter Age = 58
  - ✓ Enter Age = 65

- These five conditions will cover all the requirements for the age field.
- It is possible to use any value from each partition.

Equivalence Partitioning (Cont.)

- Ensure that the partitioning is unique and not overlapping. Each value taken should belong to only one partition.

- Total test coverage can be achieved when the test cases cover all identified partitions.

- Equivalence partitioning test coverage can be measured as the number of partitions tested divided by the total number of recognized partitions.

Drawbacks of Equivalence Partitioning

- Success of equivalence partitioning is dependent on the ability to create correct partitions.

- Ability to create partition limits based on what is mentioned in requirements. Therefore, testers who decides the test conditions will not have an understanding about the approach used by the developer.

Equivalence Partitioning - Exercise

Consider the behavior of "Order Pizza" text box given below.

Order Pizza: [_____] Submit

- Requirement of the above function.
  - Pizza values 1 to 10 are considered valid. A success message is shown.
  - Pizza values less than 1 are considered invalid and an error message will appear, "Please enter a valid count"
  - Pizza values grater than 10 are considered invalid and an error message will appear, "Only 10 Pizzas can be ordered at a time"

Perform equivalence partitioning and identify the test conditions with sample values for the above function.

Answer
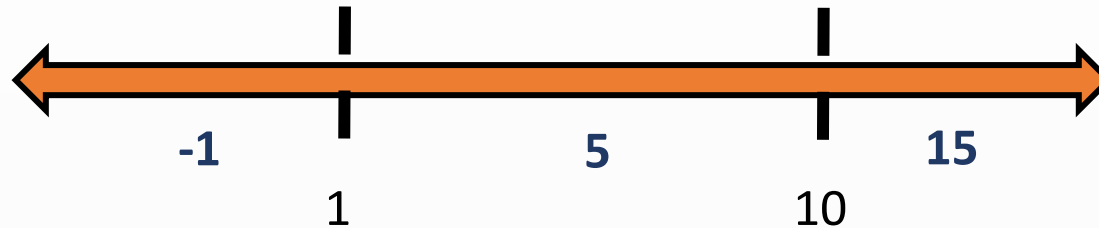
Test conditions:

- ✓ Any number less than 1 that is 0 or below is considered invalid.

  Enter Pizza Value = -1

- ✓ Numbers 1 to 10 are considered valid.

  Enter Pizza Value = 5

- ✓ Any Number greater than 10 is considered invalid.

  Enter Pizza Value = 15

Boundary Value Analysis

- A software testing technique in which tests are designed to include representatives of boundary values in a range.

- An extension of equivalence partitioning.

- Testing the boundaries of partitions.

- Useable only when the partition is ordered, consisting of numeric or sequential data.

- The minimum and maximum values of a partition are its boundary values.

Boundary Value Analysis (Cont.)

- There are high chances of finding the defects at the boundaries of a partition.

- Equivalence partitioning alone is not sufficient to catch such defects.

- Boundary Value Analysis was designed to detect anomalies at the boundaries of a partition.

How to do Boundary Value Analysis?

- Refer to the same example of Fitness1st gym form (on equivalence partitioning) where the user is required to enter the age.

- The first step of boundary value analysis is to create the equal partitions.
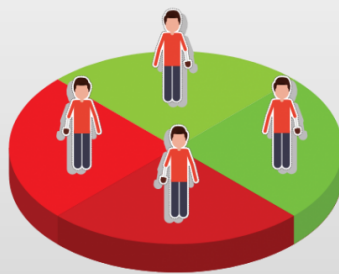
Equivalence Partitioning
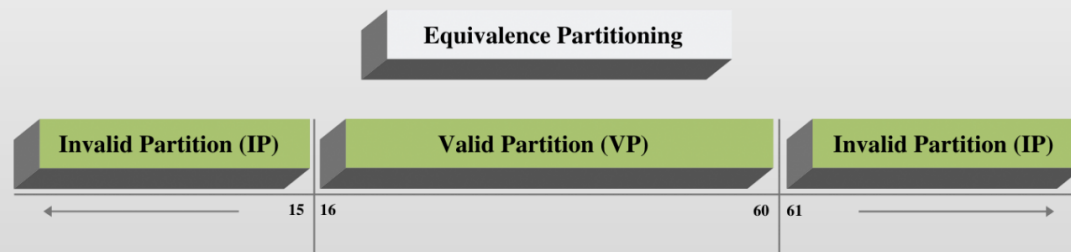
How to do Boundary Value Analysis? (Cont.)

- Concentrate on the valid partition, which ranges from 16-60.

- There is a three-step approach to identify boundaries:

  1. Identify the exact boundary value of this partition Class → 16 and 60.

  2. Get the boundary value which is one less than the exact boundary → 15 and 59.

  3. Get the Boundary Value which is one more than the precise Boundary → 17 and 61.

```
         15    17              59    61
              16                   60
```

How to do Boundary Value Analysis? (Cont.)

- Following are the combinations for boundary value for the age criteria.

  ✓ Valid Boundary Conditions : Age = 16, 17, 59, 60

  ✓ Invalid Boundary Conditions : Age = 15, 61

- Valid boundary conditions fall under valid partition class, and invalid boundary conditions fall under invalid partition class.

Boundary Value Analysis with Equivalence Partitioning

- Boundary Value Analysis is combined with equivalence partitioning to get a full set of test conditions.

    Refer to the previous example;

5   15   17   30   59   61   65
          16   60
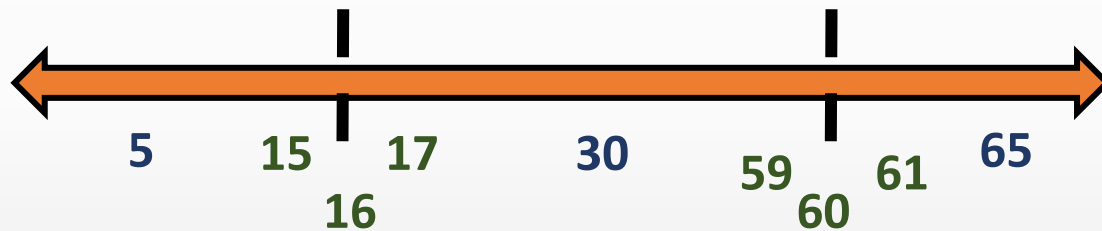
- The range is from 16 – 60;

  ➢ **Equivalence partition analysis gives test conditions as 5, 30, 65**

  ➢ **Boundary value analysis gives test conditions as 15, 16, 17, 59, 60, 61.**

- **All test conditions: 5, 15, 16, 17, 30, 59, 60, 61, 65**

Drawbacks of Boundary Value Analysis

- Boundary value and equivalence partitioning assume that the application will not allow you to enter any other characters or values. This assumption is, however, not valid for all applications.

- Boundary value analysis cannot handle situations where the decision depends on more than one input values.

  Example: If the Gym form has another field Male and Female, and the age limit will vary according to that selection.

Boundary Value Analysis- Exercise

Consider the behavior of "Order Pizza" text box given below.

Order Pizza: [   ] Submit

- Requirement of the above function.
  - o Pizza values 1 to 10 are considered valid. A success message is shown.
  - o Pizza values less than 1 are considered invalid and an error message will appear, "Please enter a valid count"
  - o Pizza values grater than 10 are considered invalid and an error message will appear, "Only 10 Pizzas can be ordered at a time"

Perform boundary value analysis and identify boundary value test conditions for the above function.

Answer
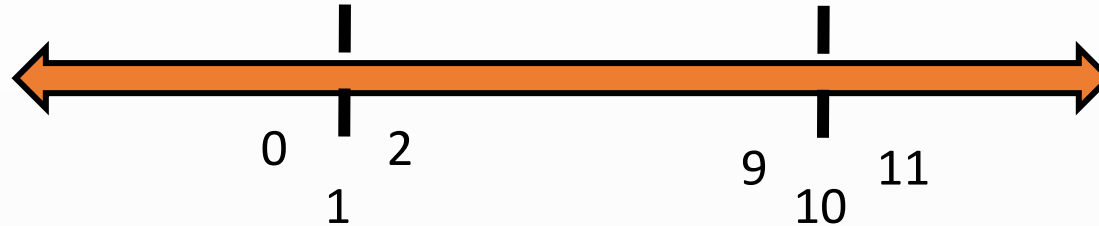
Boundary value analysis test conditions:

- ✓ Value = 0 → Invalid

- ✓ Value = 1 → Valid

- ✓ Value = 2 → Valid

- ✓ Value = 9 → Valid

- ✓ Value = 10 → Valid

- ✓ Value = 11 → Invalid

Decision Tables

- Software Testing technique in which tests are more focused on business logic or business rules.

- A decision table is a good way to deal with combinations of inputs.

- Decision tables provide a systematic way of stating complex business rules, which is useful for developers as well as for testers.

- Decision tables can be used in test design whether or not they are used in specifications, as they help testers explore the effects of combinations of different inputs and other software states that must correctly implement business rules.

How to use Decision Tables for test designing?

1. Identify a suitable function or subsystem which reacts according to a combination of inputs or events. The system should not contain too many inputs otherwise the number of combinations will become unmanageable.

   - *It is better to deal with large numbers of conditions by dividing them into subsets and dealing with the subsets one at a time.*

2. Put all identified functions into a table, listing all the combinations of True and False for each of the aspects.

Example 1: *A loan application can enter the amount of the monthly repayment or the number of years a customer wants to take to pay it back (the term of the loan). If the monthly repayment is entered by the user, then the application will process the loan amount. If the number of years is entered by the user, then the application will process the term.*

The two conditions are the loan amount and the term. Therefore, put them in a table.

Conditions
Repayment amount has been entered:
Term of loan has been Entered:

3. Identify all of the combinations (rules) of True and False.

With two conditions, each of which can be True or False, there will be four combinations.

Number of Combinations = Two to the power of the number of conditions

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:				
Term of loan has been Entered:				

How to use Decision Tables for test designing? (Contd.)

Decision table with input combinations

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been Entered:	T	F	T	F

4. Identify the correct outcome for each combination. In this example, user can enter one or both of the two fields. Each combination is also referred to as a rule.

How to use Decision Tables for test designing? (Contd.)

Decision table with input combinations with correct outcome for each combination.

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been Entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:	Y	Y		
Process term:	Y		Y	

Decision Tables for test designing

- **What happens if the customer doesn't enter anything in either of the two fields?**

- The table has depicted a combination that was not mentioned in the specification.

- Can assume that this combination should result in an error message. Therefore, add another action to the decision table.

Decision Tables for test designing (Contd.)

Decision table with additional outcomes

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been Entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:	Y	Y		
Process term:	Y		Y	
Error message:				Y

Decision Tables for test designing (Contd.)

- This highlights the strength of this technique to discover omissions and ambiguities in specifications.

- It is not unusual for some combinations to be omitted from specifications. Therefore, this is also a valuable technique to use when reviewing the test basis.

Decision Tables for test designing – Example 2

- Change in this example 1, so that the customer is not allowed to enter both repayment and term.

- Now the outcome of our table will change, because there should also be an error message if both are entered.

Example 2

Decision table for Example 2

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been Entered:	T	F	T	F
Actions/Outcomes				
Process loan amount:		Y		
Process term:			Y	
Error message:	Y			Y

Example 2

- Notice that there is only one 'Yes' for the actions in each column.

- This is called as mutually exclusive actions – only one action occurs for each combination of conditions.

Example 2

It is possible to represent the decision table in a different way by listing the actions in the cell of one row.

Conditions	Rule 1	Rule 2	Rule 3	Rule 4
Repayment amount has been entered:	T	T	F	F
Term of loan has been Entered:	T	F	T	F
Actions/Outcomes				
Result:	Error Message	Process loan amount	Process term	Error Message

Note that if more than one action results from any of the combinations, then it is better to show them as separate rows rather than combining them into one row.

How to use decision tables for test designing? (Contd.)

- The final step of this technique is to write test cases to exercise each of the four rules in our table.

Decision Tables - Exercise

A customer can use a credit card with three conditions as follows;

- New customer will get a 15% discount on all your purchases today.

- If the customer is an existing customer and holds a loyalty card, then he or she will get a 10% discount.

- If the customer has a coupon, then he or she can get 20% off today. (but it can't be used with the 'new customer' discount).

Create the decision table for above scenario.

Answer

Conditions	Rule 1	Rule 2	Rule 3	Rule 4	Rule 5	Rule 6	Rule 7	Rule 8
New Customer (15%)	T	T	T	T	F	F	F	F
Loyalty card (10%)	T	T	F	F	T	T	F	F
Coupon (20%)	T	F	T	F	T	F	T	F
Actions/Outcomes								
Discount (%)	X	X	20	15	30	10	20	0