Boundary value analysis (BVA) is based on testing the boundary values of valid and invalid partitions. The Behavior at the edge of each equivalence partition is more likely to be incorrect than the behavior within the partition, so boundaries are an area where testing is likely to yield defects.

Age>=16 and age<30

Check below video to see “Boundary Value Analysis In Software Testing

Every partition has its maximum and minimum values and these maximum and minimum values are the boundary values of a partition.

A boundary value for a valid partition is a valid boundary value. Similarly a boundary value for an invalid partition is an invalid boundary value.

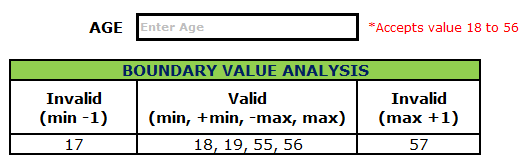
Tests can be designed to cover both valid and invalid boundary values. When designing test cases, a test for each boundary value is chosen.

For each boundary, we test +/-1 in the least significant digit of either side of the boundary.

Boundary value analysis can be applied at all test levels.

### Example on Boundary Value Analysis Test Case Design Technique:

Assume, we have to test a field which accepts Age 18 – 56

[](https://i1.wp.com/www.softwaretestingmaterial.com/wp-content/uploads/2016/03/Boundary-Value-Analysis-1.png?ssl=1)

Minimum boundary value is 18

Maximum boundary value is 56

Valid Inputs: 18,19,55,56

Invalid Inputs: 17 and 57

Test case 1: Enter the value 17 (18-1) = Invalid

Test case 2: Enter the value 18 = Valid

Test case 3: Enter the value 19 (18+1) = Valid

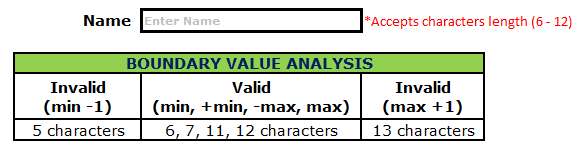
Test case 4: Enter the value 55 (56-1) = Valid

Test case 5: Enter the value 56 = Valid

Test case 6: Enter the value 57 (56+1) =Invalid

**Example 2:**

Assume we have to test a text field (Name) which accepts the length between 6-12 characters.

[](https://i0.wp.com/www.softwaretestingmaterial.com/wp-content/uploads/2016/03/Boundary-Value-Analysis-2.png?ssl=1)

Minimum boundary value is 6

Maximum boundary value is 12

Valid text length is 6, 7, 11, 12

Invalid text length is 5, 13

Test case 1: Text length of 5 (min-1) = Invalid

Test case 2: Text length of exactly 6 (min) = Valid

Test case 3: Text length of 7 (min+1) = Valid

Test case 4: Text length of 11 (max-1) = Valid

Test case 5: Text length of exactly 12 (max) = Valid

Test case 6: Text length of 13 (max+1) = Invalid

**Equivalence partitioning:**

Dividing the test input data into a range of values and selecting one input value from each range is called **Equivalence Partitioning**. This is a black box test design technique used to calculate the effectiveness of test cases and which can be applied to all levels of testing from unit, integration, system testing and so forth.

We cannot test all the possible input domain values, because if we attempted this, the number of test cases would be too large. In this method, input data is divided into different classes, each class representing the input criteria from the equivalence class. We then select one input from each class.

This technique is used to reduce an infinite number of test cases to a finite number, while ensuring that the selected test cases are still effective test cases which will cover all possible scenarios.

Let’s take a very basic and simple example to understand the *Equivalence Partitioning* concept:

If one application is accepting input range from 1 to 100, using equivalence class we can divide inputs into the classes, for example, one for valid input and another for invalid input and design one test case from each class.

In this example test cases are chosen as below:

* One is for valid input class i.e. selects any value from input between ranges 1 to 100. So here we are not writing hundreds of test cases for each value. Any one value from this equivalence class should give you the same result.
* One is for invalid data below lower limit i.e. any one value below 1.
* One is for invalid data above upper limit i.e. any value above 100.

**Boundary value analysis:**

For the most part, errors are observed in the extreme ends of the input values, so these extreme values like start/end or lower/upper values are called Boundary values and analysis of these Boundary values is called “Boundary value analysis”. It is also sometimes known as ‘range checking’.

Boundary value analysis is another black box test design technique and it is used to find the errors at boundaries of input domain rather than finding those errors in the center of input.

Equivalence Partitioning and Boundary value analysis are linked to each other and can be used together at all levels of testing. Based on the edges of the equivalence classes, test cases can then be derived.

Each boundary has a valid boundary value and an invalid boundary value. Test cases are designed based on the both valid and invalid boundary values. Typically, we choose one test case from each boundary.

Finding defects using Boundary value analysis test design technique is very effective and it can be used at all test levels. You can select multiple test cases from valid and invalid input domains based on your needs or previous experience but remember you do have to select at least one test case from each input domain.

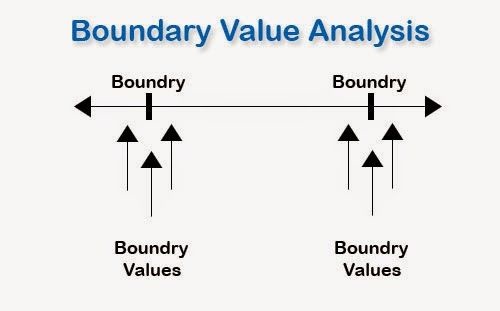
Let’s take same above example to understand the Boundary value analysis concept:

* One test case for exact boundary values of input domains each means 1 and 100.
* One test case for just below boundary value of input domains each means 0 and 99.
* One test case for just above boundary values of input domains each means 2 and 101.

**What is BVA?**

In software testing, the ***Boundary Value Analysis (BVA)*** is a ***black box test*** design technique based on test cases. This technique is applied to see if there are any ***bugs***at the boundary of the input domain. Thus, with this method, there is no need of looking for these errors at the center of this input.

BVA helps in testing the value of boundary between both valid and invalid boundary partitions. With this technique, the boundary values are tested by the creation of test cases for a particular input field.



The extreme ends or boundary partitions might depict the values of lower-upper, start-end, maximum-minimum, inside-outside etc.

In general, the BVA technique comes under the ***Stress***and ***Negative Testing***.

This technique is an easy, quick and brilliant way to catch any input errors that might occur to interrupt the functionality of a program.

So, to save their time and to cut the testing procedure short, the experts delivering ***software testing*** and quality management services rely on the Boundary Value Analysis method.

For testing of data related to boundaries and ranges, the method is considered as a very suitable one.

***An Example of Boundary Value Analysis:***

Consider the testing of a software program that takes the integers ranging between the values of -100 to +100. In such a case, three sets of the valid equivalent partitions are taken, which are – the negative range from -100 to -1, zero (0), and the positive range from 1 to 100.

Each of these ranges has the minimum and maximum boundary values. The Negative range has a lower value of -100 and the upper value of -1. The Positive range has a lower value of 1 and the upper value of 100

While testing these values, one must see that when the boundary values for each partition are selected, some of the values overlap. So, the overlapping values are bound to appear in the test conditions when these boundaries are checked.

These overlapping values must be dismissed so that the redundant test cases can be eliminated.

So, the test cases for the input box that accepts the integers between -100 and +100 through BVA are:

* *Test cases with the data same as the input boundaries of input domain: -100 and +100 in our case.*
* *Test data having values just below the extreme edges of input domain: -101 and 99*
* *Test data having values just above the extreme edges of input domain: -99 and 101*

This is a very basic example to understand the BVA testing technique!

With this technique, it is quite easy to test a small set of data in place of testing the whole lot of data sets. This is why, in software testing and quality management services**,**this method of testing is adopted more often.

***Evaluation of Boundary Value Analysis as a Software Testing Technique***

***Boundary Value Analysis Advantages:***

* *The BVA technique of testing is quite easy to use and remember because of the uniformity of identified tests and the automated nature of this technique.*
* *One can easily control the expenses made on the testing by controlling the number of identified test cases. This can be done with respect to the demand of the software that needs to be tested.*
* *BVA is the best approach in cases where the functionality of a software is based on numerous variables representing physical quantities.*
* *The technique is best at revealing any potential UI or user input troubles in the software.*
* *The procedure and guidelines are crystal clear and easy when it comes to determining the test cases through BVA.*
* *The test cases generated through BVA are very small.*

***Boundary Value Analysis Disadvantages:***

* *This technique sometimes fails to test all the potential input values. And so, the results are unsure.*
* *The dependencies with BVA are not tested between two inputs.*
* *This technique doesn’t fit well when it comes to Boolean Variables.*
* *It only works well with independent variables that depict quantity.*

In BVA, the software testers have found a fairly simple and correct testing method. This technique can be one of the most important testing techniques if used with care and correctness.

However, the technique is a little limiting when there are some issues with variable dependency or when a foresight is needed for the functionality of a system. With its inexpensive operations and more apt ways of using it, the BVA can become one of the most used testing techniques in the times to come.

But even today, many of the companies and experts delivering the software testing services**,**have adopted the BVA methods of testing.

# Boundary Value Analysis & Equivalence Partitioning with Examples

Practically, due to time and budget considerations, it is not possible to perform exhausting testing for each set of test data, especially when there is a large pool of input combinations.

* We need an easy way or special techniques that can select test cases intelligently from the pool of test-case, such that all test scenarios are covered.
* We use two techniques - **Equivalence Partitioning & Boundary Value Analysis testing techniques** to achieve this.

In this tutorial, we will learn

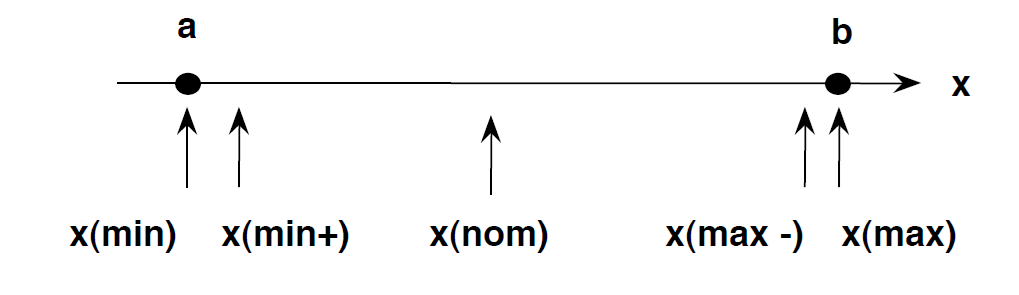
* What is Boundary Testing?
* What is Equivalent Class Partitioning?
* Example 1: Equivalence and Boundary Value
* Example 2: Equivalence and Boundary Value
* Why Equivalence & Boundary Analysis Testing

### What is Boundary Testing?

Boundary testing is the process of testing between extreme ends or boundaries between partitions of the input values.

* So these extreme ends like Start- End, Lower- Upper, Maximum-Minimum, Just Inside-Just Outside values are called boundary values and the testing is called "boundary testing".
* The basic idea in boundary value testing is to select input variable values at their:

1. Minimum
2. Just above the minimum
3. A nominal value
4. Just below the maximum
5. Maximum

[](https://www.guru99.com/images/3-2016/032316_0620_Equivalence1.png)

* In Boundary Testing, Equivalence Class Partitioning plays a good role
* Boundary Testing comes after the Equivalence Class Partitioning.

### What is Equivalent Class Partitioning?

Equivalent Class Partitioning is a black box technique (code is not visible to tester) which can be applied to all levels of testing like unit, integration, system, etc. In this technique, you divide the set of test condition into a partition that can be considered the same.

* It divides the input data of software into different equivalence data classes.
* You can apply this technique, where there is a range in the input field.

### Example 1: Equivalence and Boundary Value

* Let's consider the behavior of Order Pizza Text Box Below
* Pizza values 1 to 10 is considered valid. A success message is shown.
* While value 11 to 99 are considered invalid for order and an error message will appear, **"Only 10 Pizza can be ordered"**

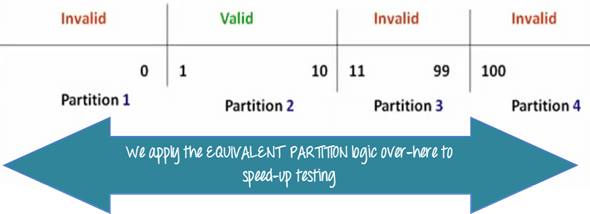
**Order Pizza:**



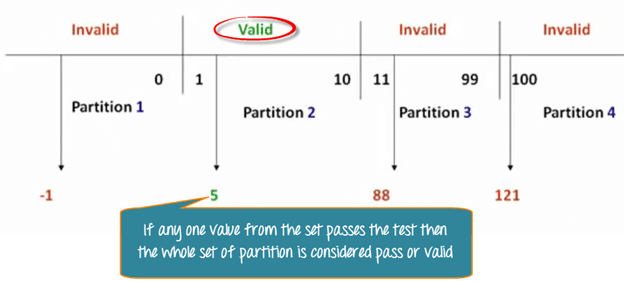
**Here is the test condition**

1. Any Number greater than 10 entered in the Order Pizza field(let say 11) is considered invalid.
2. Any Number less than 1 that is 0 or below, then it is considered invalid.
3. Numbers 1 to 10 are considered valid
4. Any 3 Digit Number say -100 is invalid.

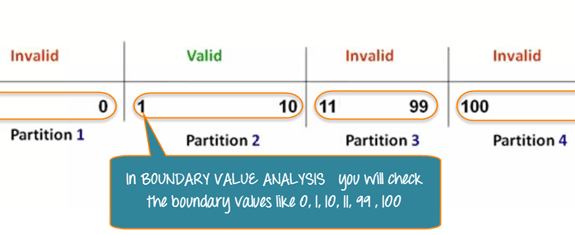
We cannot test all the possible values because if done, the number of test cases will be more than 100. To address this problem, we use equivalence partitioning hypothesis where we divide the possible values of tickets into groups or sets as shown below where the system behavior can be considered the same.

[](https://www.guru99.com/images/3-2016/032316_0620_Equivalence4.png)

The divided sets are called Equivalence Partitions or Equivalence Classes. Then we pick only one value from each partition for testing. The hypothesis behind this technique is **that if one condition/value in a partition passes all others will also pass**. Likewise**, if one condition in a partition fails, all other conditions in that partition will fail**.

[](https://www.guru99.com/images/3-2016/032316_0620_Equivalence5.png)

**Boundary Value Analysis**- in Boundary Value Analysis, you test boundaries between equivalence partitions

[](https://www.guru99.com/images/3-2016/032316_0620_Equivalence6.png)

In our earlier example instead of checking, one value for each partition you will check the values at the partitions like 0, 1, 10, 11 and so on. As you may observe, you test values at**both valid and invalid boundaries**. Boundary Value Analysis is also called**range checking**.

Equivalence partitioning and boundary value analysis(BVA) are closely related and can be used together at all levels of testing.

### Example 2: Equivalence and Boundary Value

Following password field accepts minimum 6 characters and maximum 10 characters

That means results for values in partitions 0-5, 6-10, 11-14 should be equivalent

**Enter Password:**



|  |  |  |
| --- | --- | --- |
| **Test Scenario #** | **Test Scenario Description** | **Expected Outcome** |
| 1 | Enter 0 to 5 characters in password field | System should not accept |
| 2 | Enter 6 to 10 characters in password field | System should accept |
| 3 | Enter 11 to 14 character in password field | System should not accept |

### Examples 3: Input Box should accept the Number 1 to 10

Here we will see the Boundary Value Test Cases

|  |  |
| --- | --- |
| **Test Scenario Description** | **Expected Outcome** |
| Boundary Value = 0 | System should NOT accept |
| Boundary Value = 1 | System should accept |
| Boundary Value = 2 | System should accept |
| Boundary Value = 9 | System should accept |
| Boundary Value = 10 | System should accept |
| Boundary Value = 11 | System should NOT accept |

### Why Equivalence & Boundary Analysis Testing

1. This testing is used to reduce a very large number of test cases to manageable chunks.
2. Very clear guidelines on determining test cases without compromising on the effectiveness of testing.
3. Appropriate for calculation-intensive applications with a large number of variables/inputs

**Summary:**

* Boundary Analysis testing is used when practically it is impossible to test a large pool of test cases individually
* Two techniques - Equivalence Partitioning & Boundary Value Analysis testing techniques are used
* In Equivalence Partitioning, first, you divide a set of test condition into a partition that can be considered.
* In Boundary Value Analysis you then test boundaries between equivalence partitions
* Appropriate for calculation-intensive applications with variables that represent physical quantities