

Lecture - 48
Equivalence Class Testing - 1

Welcome we will continue our discussion on Equivalence Class Testing. Last time we had just discussed little bit about equivalence class testing and will continue from there.

(Refer Slide Time: 00:30)

Equivalence Class Partitioning

- The input values to a program:
 - Partitioned into **equivalence classes**
- Partitioning is done such that:
 - **Program behaves in similar ways to every input value belonging to an equivalence class.**
 - **At the least, there should be as many equivalence classes as scenarios.**

The slide includes a flowchart on the right with nodes and arrows, and a use case diagram with a red arrow labeled 'Start use case' and 'end use case'. The bottom of the slide features the IIT Kharagpur and NPTEL logos.

The main idea behind equivalence class testing is that, we partition the data value. The data that is input to a function we model it, and partition it into equivalence classes. Now, let me just tell little bit about, why we want to partition it into equivalence classes. And how does it help in designing test cases. We partition it into equivalence classes, because each data point in a equivalence class exhibit similar behavior. If, we think of this as a representation of a program the control flow in a program, can see that there are many ways the control can flow and each way the control flows represents a behavior.

So, if we partition the data that is input to a program, if this is the set of data. If this is the set of data that is to be input to the program and we know that these are the data for which it follows this path. And these are the data for which it follows this path. So, these represent different behavior, different output, different type of processing and so on. And may be this one corresponds to a behavior exhibited by this and so on.

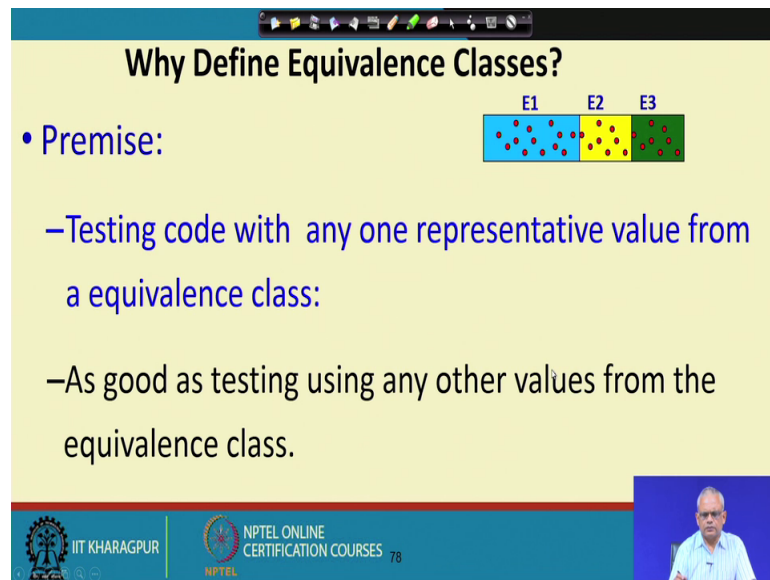
Now all the data points here, they correspond to the behavior expressed by this processing. So, the idea here between equivalence class partitioning is that, if we can identify the equivalence classes, for which the behavior, for any class is similar for all the data points, then we can just test it with one of the data point here. So, that we know that these behavior is we will also select a data point that will drive it through a different behavior and so on.

That is the main idea here that a program behaves in similar ways to every input value belonging to an equivalence class. So, this is a equivalence class, then every input will exhibit similar behavior, because the set of statements that are getting executed are similar. And therefore, if it works for one point here where it followed this set of statement execution, then it is reasonable to expect, that it will also work satisfactorily for all other points in that same equivalence class. But we also need to check about the behavior when it follows this path and so on this set of statements, but one thing that we must remember is that from a black box description of a functionality, that is a huge case.

We know that there are alternate scenarios. In the alternate scenarios, this the main line scenario and then there are many alternate scenarios. We can think of that in the main line scenario, certain set of statements here this the main line scenario. And corresponding to the main line scenario certain set of statements get executed. And for alternate scenarios different set of statements get executed. And therefore, if we look at the huge case description and look at the main line scenario and the alternate scenarios we can associate equivalence classes to each of this.

The main line scenario the data that makes the system execute the main line scenario corresponds to one equivalence class and for every alternate path we have a different equivalence class. This is one way to identify the equivalence classes, but will see that even when it follows the main line scenario, there may be further equivalence classes in that we will just explore that.

(Refer Slide Time: 05:52)



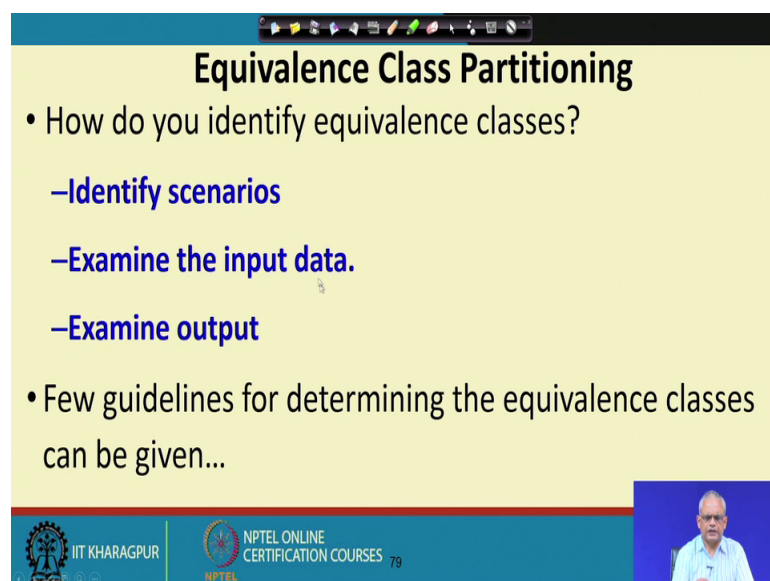
Why Define Equivalence Classes?

- **Premise:**
 - Testing code with any one representative value from a equivalence class:
 - As good as testing using any other values from the equivalence class.

The slide features a diagram with three columns labeled E1, E2, and E3. E1 contains blue dots, E2 contains yellow dots, and E3 contains green dots. The footer includes the IIT Kharagpur logo, NPTEL Online Certification Courses logo, and the number 78. A small video inset of the presenter is visible in the bottom right corner.

So, let me just repeat here, that the total data input data domain. The input data domain we model it and then we split it into equivalence classes such that, the tester or we if we are doing the testing the tester feels that for every data point here, the system exhibit similar behavior. And, then since all of these make the system execute certain set of statements it is reasonable to execute using any one of these points. We can select any one of these points and similarly with other equivalence classes and that will form our equivalence test cases equivalence class test cases.

(Refer Slide Time: 07:07)



Equivalence Class Partitioning

- How do you identify equivalence classes?
 - **Identify scenarios**
 - **Examine the input data.**
 - **Examine output**
- Few guidelines for determining the equivalence classes can be given...

The slide footer includes the IIT Kharagpur logo, NPTEL Online Certification Courses logo, and the number 79. A small video inset of the presenter is visible in the bottom right corner.

One way of identifying the equivalence classes is by examining the use case description. The main line scenario and alternate scenarios correspond to different equivalence classes, but beyond that we said that every equivalence class there may also consist of other equivalence classes, how do we identify that?

One is based on the use case diagram and huge case description, we identify the scenarios, we examine the input data which make it execute a specific scenario and those set of input data constitute the equivalence class. But, then there are other guidelines for identifying equivalence classes, because this allows us to identify a broad set of equivalence classes, but let us see, what are the other guidelines?

(Refer Slide Time: 08:13)

• If an input is a range, one valid and two invalid equivalence classes are defined.
Example: 1 to 100

• If an input is a set, one valid and one invalid equivalence classes are defined. Example: {a,b,c}

• If an input is a Boolean value, one valid and one invalid class are defined.

Example:

- **Area code:** input value defined between 10000 and 90000---
- **Password:** string of six characters ---

Guidelines to Identify Equivalence Classes

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 80

The other guidelines are that we look at the input values. If, the input is a range of values as we find the description of the functional requirements or the use case. If, the input corresponds to a range of values, let us say 1 to 100, if these are set of valid values, then we have 3 equivalence classes; one is the valid set of equivalence classes and there are 2 invalid equivalence classes on the both ends of the valid equivalence class. So, this is the invalid equivalence class this a invalid equivalence class and this is a valid equivalence class.

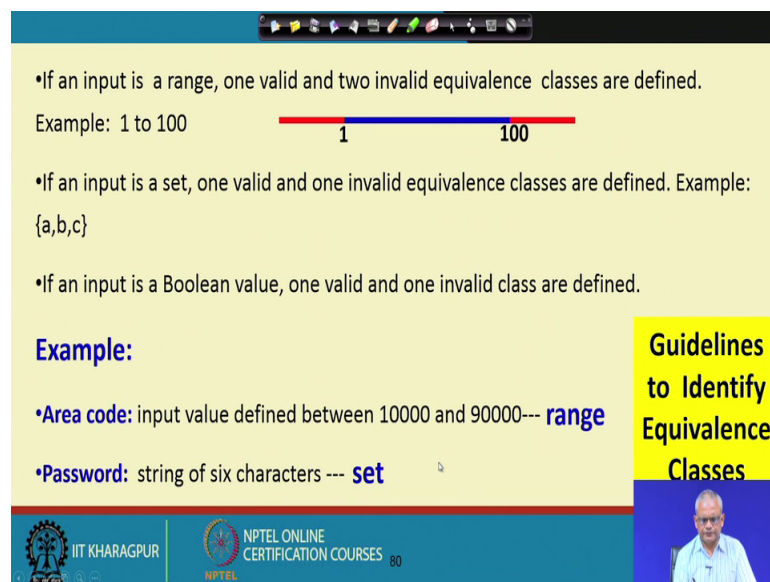
Now, what if the input is a set of values? Let us say only a, b, c let us say our system our software takes only a character and based on the character it does certain functions these are the menu choices let us say a b c. And these are only the valid menu choices you

have to input any one of these. In that case this defines a valid set of equivalence class and anything other than a b c these are the invalid set of equivalence class.

So, if the input is a set we need to identify a valid set of equivalence class and invalid set of equivalence class. What, if the input is a Boolean value, again here we have one valid that is the Boolean value and any other value such as a character control characters and so on these will constitute the invalid class? Now, let us see let us check our understanding for 2 example problems. Let us say a function takes an area code that is the pin code. And the pin code is let us say has value between 10, 000 and 90, 000. So, what are the equivalence classes here? We can see here that this defines a range 10, 000 to 90, 000.

And therefore, there will be 3 equivalence classes; one is the valid equivalence class between 10 000 and 90 000 and another less than 10 000 another more than 90 000. So, the crucial thing to recognize that for the area code or the pin code, it denotes a range of values and as soon as we know that it is a range of values, we can identify 3 equivalence classes, which is valid and 2 invalid set of equivalence classes.

(Refer Slide Time: 11:24)



•If an input is a range, one valid and two invalid equivalence classes are defined.
Example: 1 to 100

•If an input is a set, one valid and one invalid equivalence classes are defined. Example: {a,b,c}

•If an input is a Boolean value, one valid and one invalid class are defined.

Example:

- Area code: input value defined between 10000 and 90000--- **range**
- Password: string of six characters --- **set**

Guidelines to Identify Equivalence Classes

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 80

Now, let us look at another one a function let us say takes a password. Let us say the function is log in and it takes a password and the password consists of a string of 6 characters. Now, what will be the equivalence classes here? It is crucial to recognize that string of 6 characters is actually a set; it is a finite set of all possible string of 6

characters. And therefore, as soon as we recognize that it is a set will identify 2 equivalence classes in this case, that is one is a valid equivalence class, which is a member of the set.

And another is invalid equivalence class, which is less than 6 characters more than 6 character etcetera etcetera. So, those are the invalid. So, there are 2 equivalence classes here one is the valid equivalence class, which is a member of the set and invalid equivalence class all the data points which are not the member of the set.

(Refer Slide Time: 12:57)

The slide is titled "Equivalent class partition: Example". It contains a bulleted list of triangle types: "Isosceles", "Scalene", and "Equilateral, etc.". To the right of the list, there is a handwritten function signature in blue ink: `determineTriangle(int side1, int side2, int side3)`. The slide also features a small video inset of a speaker in the bottom right corner and logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES at the bottom.

Now, let us try to solve some problems. Let us say we have a function which takes 3 integers. And, then determines what is the type of the triangle, that it forms may be the name of the function is determine triangle type; determine triangle type. And, it takes 3 integers side 1, side 2, and side 3. Now, based on whether the triangle defined by this 3 sides, they form a isosceles, scalene, equilateral etcetera. It defines the type of the triangle, otherwise if it is not a valid triangle it will display valid invalid triangle.

How do we identify the equivalence classes here? By observing the output here, the display whether it is isosceles scalene or equilateral, we know that these correspond to different scenarios. If we give set of values set of 3 values, which form a isosceles then this corresponds to scenario scalene equilateral etcetera. And therefore, by observing the output we can get an idea of the scenarios and equivalence classes that these will define.

(Refer Slide Time: 15:01)

Equivalent class partition: Example

- Given three sides, determine the type of the triangle:
 - Isosceles
 - Scalene
 - Equilateral, etc.
- Hint: scenarios correspond to output in this case.

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES

So, for this problem the initial set of equivalence classes are the scenarios, which correspond to the output in this case.

(Refer Slide Time: 15:15)

Equivalence Partitioning

- First-level partitioning:
 - Valid vs. Invalid test cases

$f(\text{int } i)$

data

Valid Invalid

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 82

But, let us see if we have any general problem a function let us say f , where the function f which takes a parameter may be integer float or something character. Let me just write integer i f is a function, which takes an integer or float or something, then we can say that to start with will have 2 equivalence classes. The set of data can be partitioned into 2 equivalence classes.

So, these are all the set of data, and we can partition the data into 2 equivalence classes; one is the valid set of equivalence classes sorry valid set of data which is the valid equivalence class and the invalid set of data, which is the invalid equivalence class. So, the 2 equivalence class for every problem we should be able to identify to start with and then as we proceed to find further equivalence classes, will find further valid set of equivalence classes, further valid set of equivalence classes and further invalid set of equivalence classes. Let us see, how we go about doing that?

(Refer Slide Time: 16:46)

Equivalence Partitioning

- Further partition valid and invalid test cases into equivalence classes

The diagram shows a large circle divided vertically into two halves. The left half is labeled 'Valid' and the right half is labeled 'Invalid'. Inside the 'Valid' half, there are several yellow, irregular shapes of various sizes and orientations. Inside the 'Invalid' half, there are also several yellow, irregular shapes of various sizes and orientations. The shapes are scattered throughout both halves, representing test cases that have been partitioned into equivalence classes.

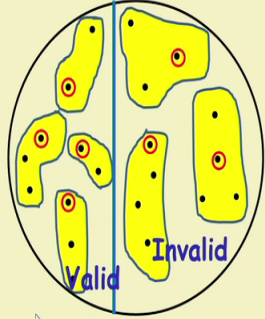
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 83

So, this says that to start with the input data will partition it into valid and invalid set of equivalence classes. And, in the next step will further identify equivalence classes within the valid set, and also equivalence classes within the invalid set.

(Refer Slide Time: 17:18)

Equivalence Partitioning

- Create a test case using at least one value from each equivalence class



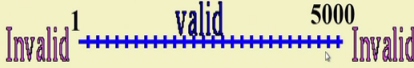
IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 84

And, once we have done that of course, within the equivalence classes, we might for some problems find further equivalence classes here, but once we have ultimately found the equivalence classes. Then, we just pick one data value from each equivalence class and that forms our equivalence class test cases.

(Refer Slide Time: 17:39)

Equivalence Class Partitioning

- If the input data to the program is specified by a **range of values**:
 - e.g. numbers between 1 to 5000.
 - One valid and two invalid equivalence classes are defined.



IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 85

Now, let us look at some problems and see how to identify the equivalence class test cases? Let us say our function takes a integer between 1 to 5000 these are the valid set of data any data any integer value that is between 1 to 5000 is a valid input. So, this

corresponds to a range of input and we can clearly identify that there is a valid class. And, there are 2 invalid classes less than one and more than 50, 000.

(Refer Slide Time: 18:28)

Equivalence Class Partitioning

- If input is an enumerated set of values, e.g. :
 - {a,b,c}
- Define:
 - One equivalence class for valid input values.
 - Another equivalence class for invalid input values..

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 86

Similarly, if we have a set of values like a, b, c, we find one equivalence class which is valid, and another which is invalid and then we will pick up one value here like b or c or something and something, which is not part of here may be f or something.

(Refer Slide Time: 18:51)

Example

- A program reads an input value in the range of 1 and 5000:
 - Computes the square root of the input number

$i \rightarrow \text{SQRT} \rightarrow \sqrt{i}$

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 87

Now, let us look at another example here the program reads an input value in the range of 1 to 5000 and computes the square root of the input number. The input is a range here

can easily identify. And therefore, to start with there are 3 classes 1 valid class and 2 invalid classes, 1 invalid class less than 1 another invalid class more than 5000.

(Refer Slide Time: 19:23)

Example (cont.)

- Three equivalence classes:
 - The set of negative integers,
 - Set of integers in the range of 1 and 5000,
 - Integers larger than 5000.

Invalid 1 valid 5000 Invalid

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 88

The slide features a yellow background with a blue header and footer. A diagram shows a horizontal number line with a blue dashed line between 1 and 5000 labeled 'valid'. The regions to the left of 1 and to the right of 5000 are labeled 'Invalid' in purple. A small video inset of a speaker is in the bottom right corner.

So, the set of negative integers, the set of integers between 1 and 5000 and integers larger than 5000 these are the 3 equivalence classes.

(Refer Slide Time: 19:40)

Example (cont.)

- The test suite must include:
 - Representatives from each of the three equivalence classes:
 - A possible test suite can be: $\{-5, 5000, 6000\}$.

Invalid 1 valid 5000 Invalid

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 89

The slide features a yellow background with a blue header and footer. A diagram shows a horizontal number line with a blue dashed line between 1 and 5000 labeled 'valid'. The regions to the left of 1 and to the right of 5000 are labeled 'Invalid' in purple. A small video inset of a speaker is in the bottom right corner.

Then, we can pick representative values from each, we may choose minus 5 500 6000 or you may choose any other value, but then once we form the test cases we assume that if

it passes value a valid value like 500, than any other value it will also pass there is the assumption behind equivalence class testing.

(Refer Slide Time: 20:08)

Equivalence Partitioning

- **A set of input values constitute an equivalence class if the tester believes that these are processed identically:**
 - Example : `issue book (book id)` }
 - Different set or sequence of instructions may be executed based on book type.

```
graph TD; Book[Book] --> Issue[Issue book]; Book --> Reference[Reference book]; Issue --> Single[Single volume]; Issue --> Multiple[Multiple volume];
```

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES | 90

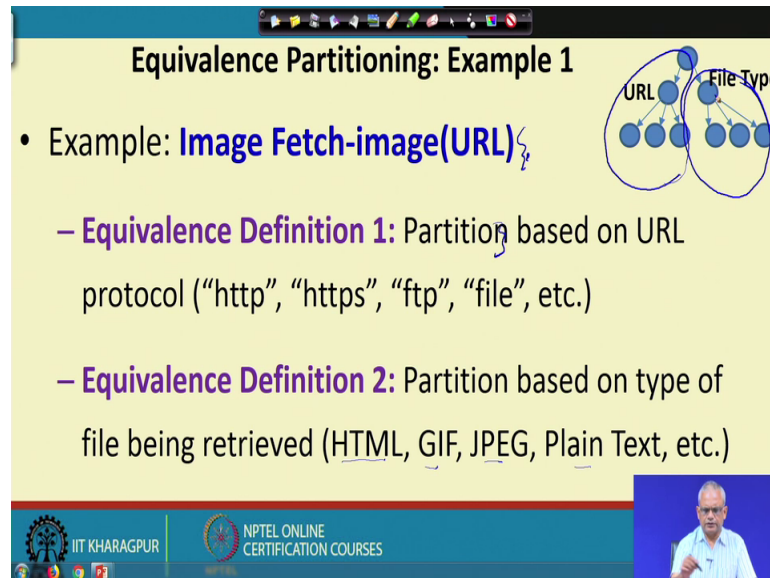
So, we can say that a set of input values constitute an equivalence class. If the tester believes that these are processed identically by the program, let me repeat that. A set of input values constitute an equivalence class if the tester believes that these are processed identically. Let us do some more problems to get a clear idea about, how to proceed to identify the equivalence classes? Let us assume that, we have a function issue book for a library software and then it takes the book id and then issues that book.

But, then we can imagine that the code that is there in the issue book, depending on the type of the book, it will execute different set of statements. For example, if the book id is a reserved book, then it will execute some set of statements and it may output, that the book is reserved cannot be issued out.

Similarly, the book is a reference book, than it will also execute slightly different set of instructions. And, then it will say that it is a reserved book cannot be issued. If, it is a book that can be issued, it will issue it, if it is a let us say a multi volume book a set of books. Then it will say that these many number of books are issued out. So, based on the input we can identify the equivalence classes.

The first level, we can have the issue book first is of course, the valid and invalid data something, which is not a book id may be a floating point number or something. So, valid and invalid and then for the valid book, we can have a issuable book and a reference book. And for the issuable book we can have single volume and multiple volume.

(Refer Slide Time: 22:52)



Equivalence Partitioning: Example 1

- Example: **Image Fetch-image(URL)**
- **Equivalence Definition 1:** Partition based on URL protocol (“http”, “https”, “ftp”, “file”, etc.)
- **Equivalence Definition 2:** Partition based on type of file being retrieved (HTML, GIF, JPEG, Plain Text, etc.)

The slide includes a diagram of a tree structure with nodes. The left side of the tree is circled and labeled 'URL', and the right side is circled and labeled 'File Type'. The bottom of the slide features logos for IIT KHARAGPUR and NPTEL ONLINE CERTIFICATION COURSES, along with a small video inset of a speaker.

Let us do some more examples, let us say we have a function, which is fetch image it takes a URL and then fetches the image and returns the image. So, the function, it takes 1 parameter which is the URL and depending on the image present there, it retrieves the image and returns the image. Now, how do we define the equivalence class here? One is that based on the type of URL. For example, it may be using http protocol http’s protocol ftp file. So, this is the protocol used in the URL based on that we can partition into a set of equivalence classes.

So, that I have written here these the set of data valid data and then based on the URL specified, we can have some equivalence classes like http ftp file http s etcetera, but again the specific URL may point to different types of images. So, depending on the image type we again can define a equivalence class, depending on whether it is a HTML image, GIF image, JPEG, Plain Text etcetera.

So, we have another set of equivalence class here. Just want to make this point that based on the input it is not that there is a single hierarchy, we can have multiple hierarchy

based on different characteristics of the input the URL the protocol of the URL the file type and so on. So, even though the concept is straight forward, but sometimes identifying the equivalence classes can be requiring a some thought on the input data value.

(Refer Slide Time: 25:31)

Equivalence Partitioning: Single Parameter Function

- `issue-book(int book-id)`
- Create a test case for at least one value from each equivalence class

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 92

Once, we identify the equivalence classes, then we just select one data point corresponding to the valid equivalence classes. So, that is single volume multiple volume and reference. And, similarly we select one data point for the invalid equivalence classes.

(Refer Slide Time: 25:59)

Multiparameter Functions

- `postGrade(Roll, CourseNo, Grade)`

IIT KHARAGPUR | NPTEL ONLINE CERTIFICATION COURSES 93

So, far we just looked at how to identify the equivalence classes, when a function takes one parameter, but we can have a function which takes multiple parameters. Now, how do we identify the equivalence classes here. Of course, it is rather straight forward of the discussion that we had so far we will examine each input parameter, and based on the input parameter, we can identify the equivalence classes for the different input parameters. But, now how do we define the test cases? We can select points from here for corresponding to this parameter points corresponding to this parameter and so on.

But, the test for the test case one way is that we take one point from here and form the test case. Another way can be that we take one point here and corresponding to that we define test cases by considering each of the point here and each of the point here. So, that will give us large number of test cases. So, let me just repeat that once we identify the equivalence classes for each of the 3 parameters, one way is that we define the test cases such that every equivalence class is represented in our test case. The other way is that we form a (Refer Time: 27:53) combination of the input values. Now, we are almost at the end of this lecture, we will continue from this point in the next lecture.

Thank you.