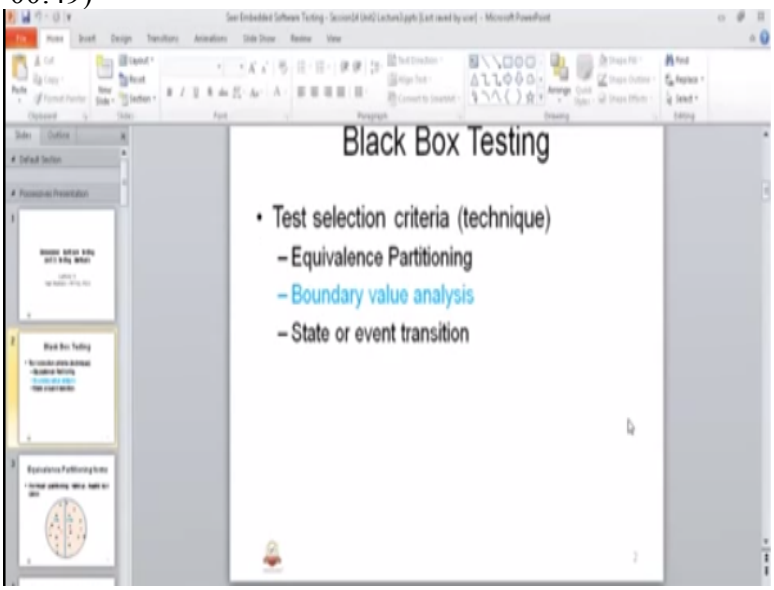
Embedded Software Testing
Unit 2: Testing Methods
Lecture 14
Seer Academy- NPTEL MOU

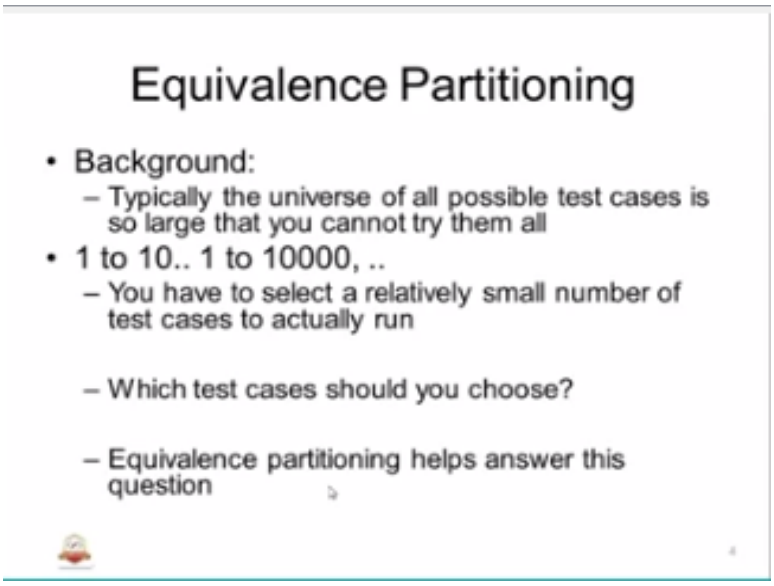
Welcome to the next session of embedded software testing unit to the testing method lecture-14 these important unit because, based on this the entire embedded testing methods of alive, so will be giving a detail understanding and detail example through of different testing methods, very important to understand and testing methods are quietly spoke about black box testing, and different techniques for the selection techniques of black box testing

(Refer Slide Time: 00:49)

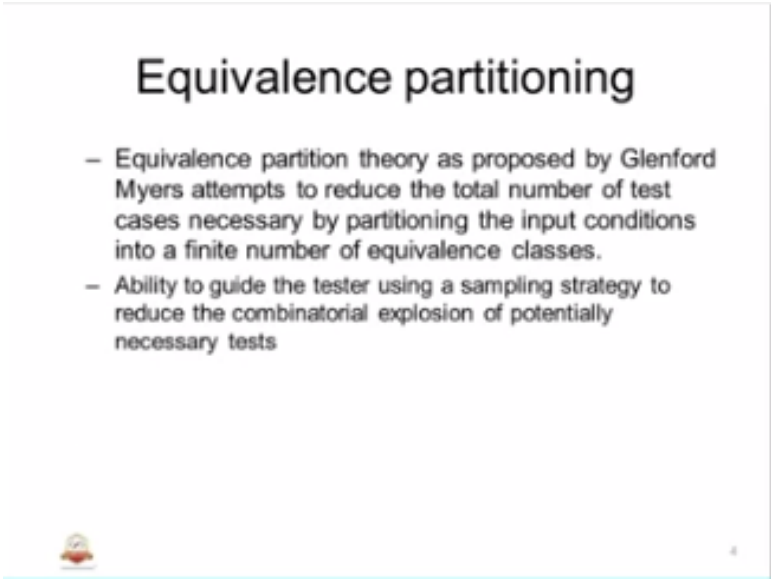


Today we will study about value analysis, previous a in different techniques and criteria techniques again equivalence platform is, what are the fundamentals that we had a understood any equivalence partitioning in different courses, electronics being fundamentally, values are comes in to different links okay, so before that we will discuss a glands of what we study in a equivalence partitioning..

(Refer Slide Time: 01:28)

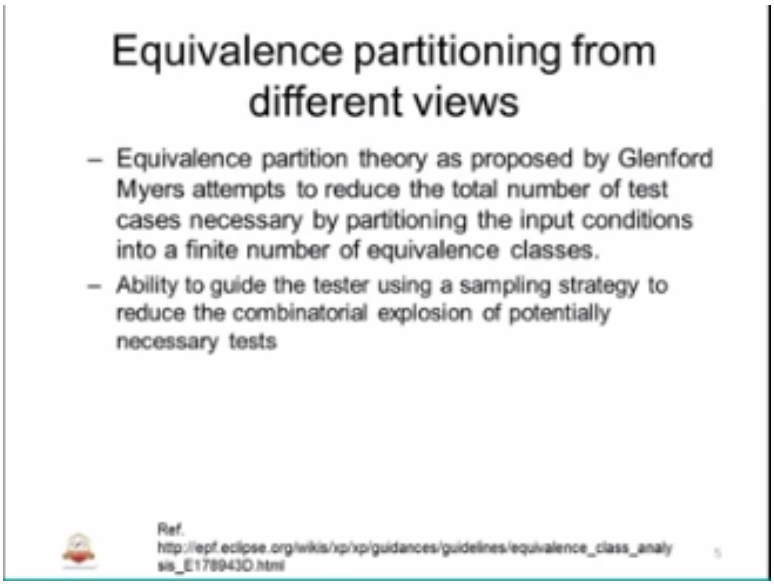


(Refer Slide Time: 02:09)



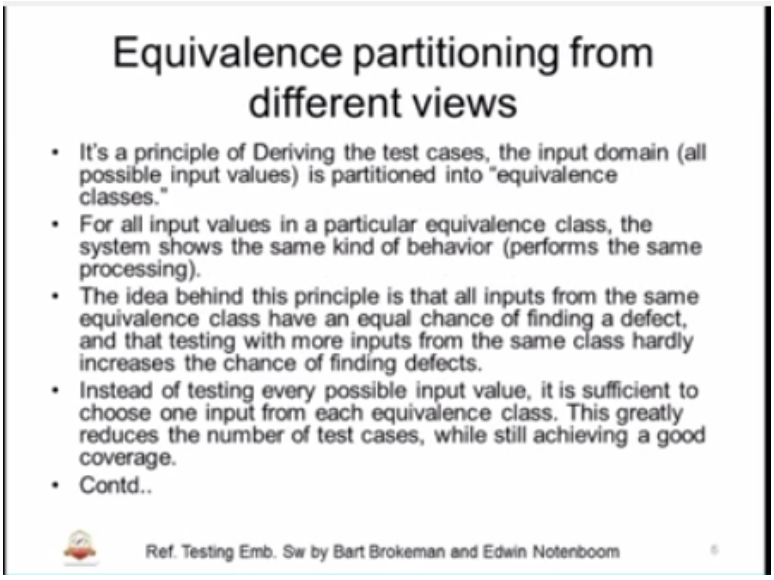
Equivalence partitioning we know why we need, because we cannot effort to have a humorous test cases, we have a possibility of doing test which could be a different number of inputs, because instruction either or normal way for a typically regalement we required test only the submission levels of inputs, how we can do that reduction if by having the partitioning that is called equivalence partitioning..

(Refer Slide Time: 02:10)



Also we understood that equivalence partitioning reduce the total number of test, it partitioning the input condition, in to the finite number of equivalence classes.

(Refer Slide Time: 02:22)



(Refer Slide Time: 02:22)

Equivalence partitioning contd.

- It's a principle of Deriving the test cases, the input domain (all possible input values) is partitioned into "equivalence classes."
- For all input values in a particular equivalence class, the system shows the same kind of behavior (performs the same processing).
- The idea behind this principle is that all inputs from the same equivalence class have an equal chance of finding a defect, and that testing with more inputs from the same class hardly increases the chance of finding defects.
- Instead of testing every possible input value, it is sufficient to choose one input from each equivalence class. This greatly reduces the number of test cases, while still achieving a good coverage.
- Contd..

Ref. Testing Emb. Sw by Bart Brokeman and Edwin Notenboom

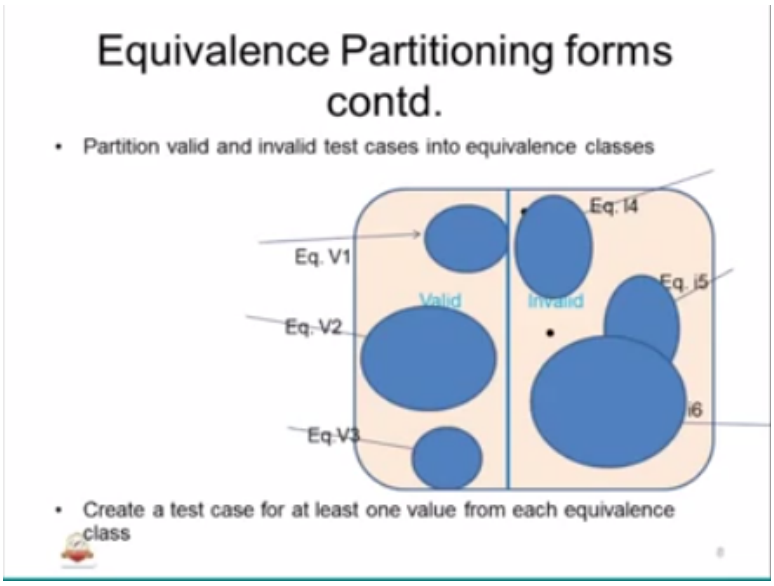
(Refer Slide Time: 02:22)

Equivalence Partitioning forms

- It's a principle of Deriving the test cases, the input domain (all possible input values) is partitioned into "equivalence classes."
- For all input values in a particular equivalence class, the system shows the same kind of behavior (performs the same processing).
- The idea behind this principle is that all inputs from the same equivalence class have an equal chance of finding a defect, and that testing with more inputs from the same class hardly increases the chance of finding defects.
- Instead of testing every possible input value, it is sufficient to choose one input from each equivalence class. This greatly reduces the number of test cases, while still achieving a good coverage.
- Contd..

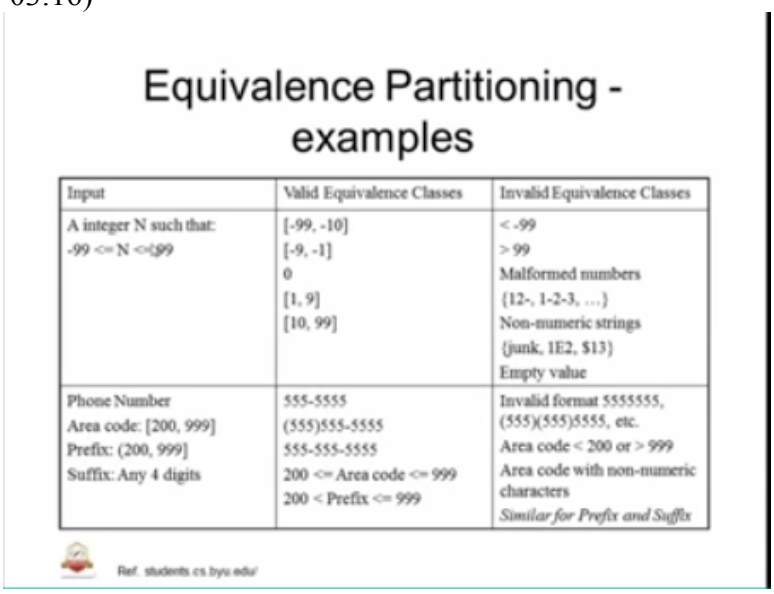
Ref. Testing Emb. Sw by Bart Brokeman and Edwin Notenboom

(Refer Slide Time: 02:23)



We also studied about the first level of valid and invalid testing, first we are going to define all the testing's and we are going to have a partitions of one usual behavior what is the expected for testing those inputs, those who are called valid equivalence partitioning, the other than which is normal behavior in terms of outside the cont. Or whatever it is those who are called equivalence partitioning, and we define the numbers accordingly, EQ v1,EQ v2, EQ v3, base of the classes and each class will have one of each case required form.

(Refer Slide Time: 03:16)



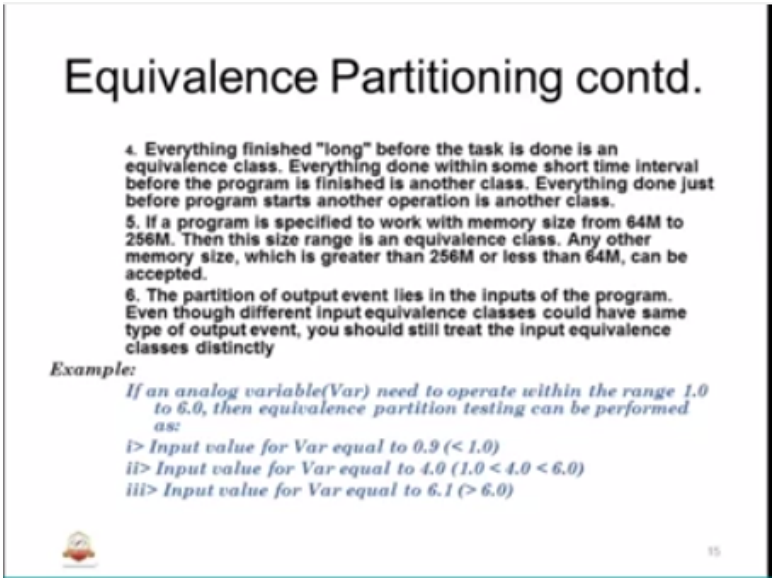
Equivalence Partitioning - examples

Input	Valid Equivalence Classes	Invalid Equivalence Classes
A integer N such that: -99 <= N <= 99	[-99, -10] [-9, -1] 0 [1, 9] [10, 99]	< -99 > 99 Malformed numbers {12-, 1-2-3, …} Non-numeric strings {junk, 1E2, $13} Empty value
Phone Number Area code: [200, 999] Prefix: (200, 999] Suffix: Any 4 digits	555-5555 (555)555-5555 555-555-5555 200 <= Area code <= 999 200 < Prefix <= 999	Invalid format 5555555, (555)(555)5555, etc. Area code < 200 or > 999 Area code with non-numeric characters *Similar for Prefix and Suffix*

Ref. students.cs.byu.edu/

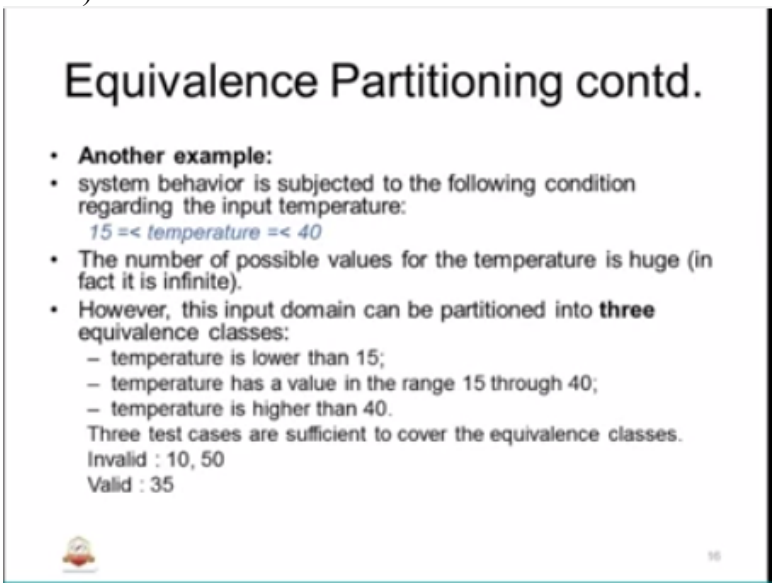
We also study an example of integrin and ranging from 99 to <N<<+99 so valid equivalence classes (-99+1) like this we have a equivalence classes so regularly we have about 3 to 4 invalid equivalence classes, were we trying to input the values so they were to testing the values in arranging of inputs, so I will tell you an another example of number which we have studied,

(Refer Slide Time: 03:55)



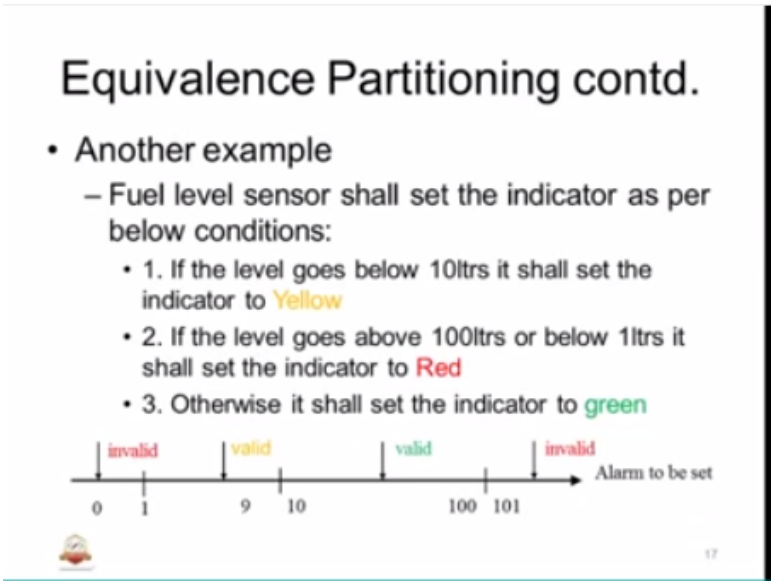
And also we have understood about three or four guide lines that are about three or four, five guide lines that are the important in terms of time, memory and size, range, count, etc….

(Refer Slide Time: 04:07)



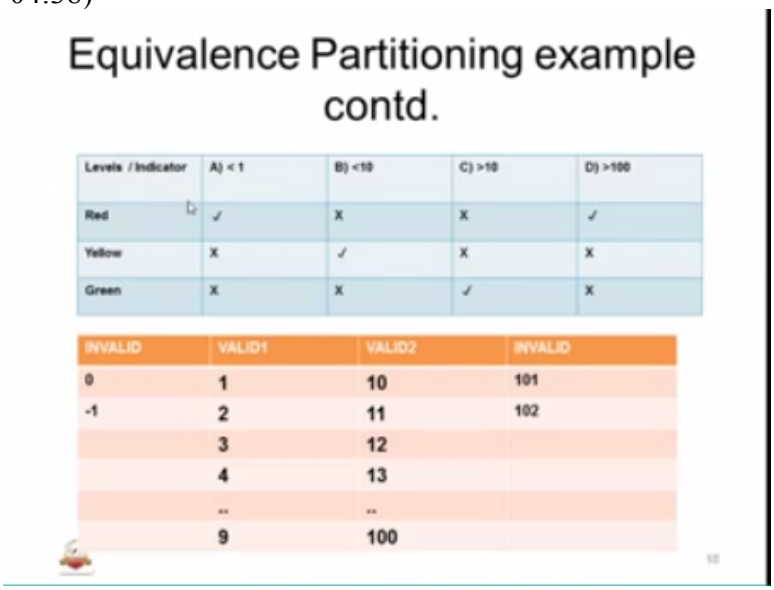
We took few more example of temperature 15=< temperature=< 40

(Refer Slide Time: 04:17)



And then we have a fuel level sensor, having three types of indicator yellow, red and green, so we know that, what is invalid? What is valid? So basically we are going to draw a range of all the equivalence,

(Refer Slide Time: 04:38)



And A is the range we are going to create a table is called indicator table having all the possible complications defined for each of that inputs with the help of different columns, the first level we are going to have a type of classes divided as valid, invalid zero table

(Refer Slide Time: 04:59)

Equivalence Partitioning example contd.

- **Output equivalence:**
- Equivalence partitioning can also be applied to the output domain of the system. (output based testing)

INVALID	VALID1	VALID2	INVALID
0	1	10	101
-1	2	11	102
	3	12	
	4	13	
	..	..	
	9	100	

(Refer Slide Time: 05:02)

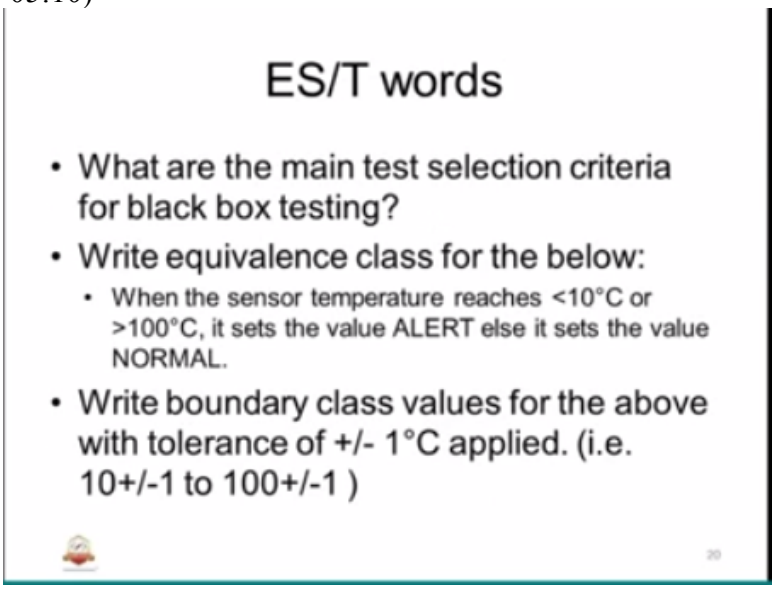
Equivalence Partitioning contd.

- **Output equivalence:**
- Equivalence partitioning can also be applied to the output domain of the system. (output based testing)
- Test cases are then derived that cover all equivalent output classes.

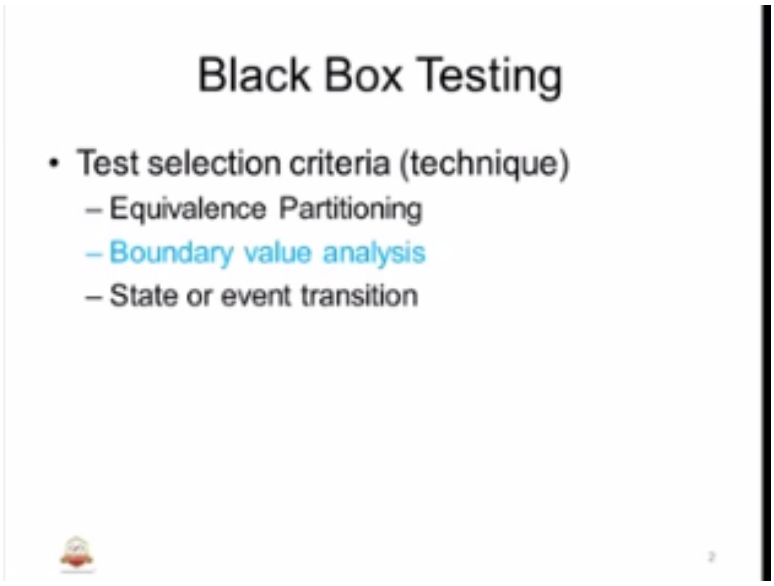
We also know that we can have the output equivalence classes, defined as well not only the input in equivalence

(Refer Slide Time: 05:10)

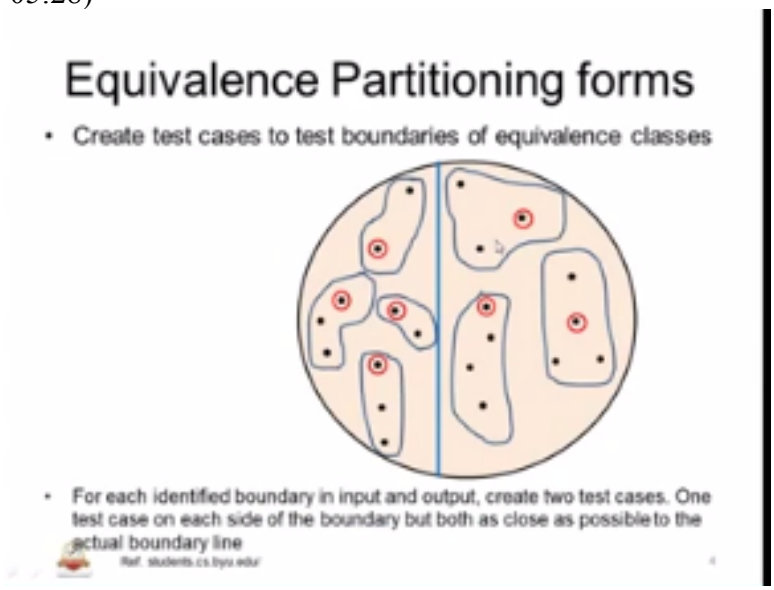
(Refer Slide Time: 05:10)



(Refer Slide Time: 05:12)
So that is about equivalence partitioning
(Refer Slide Time: 05:17)



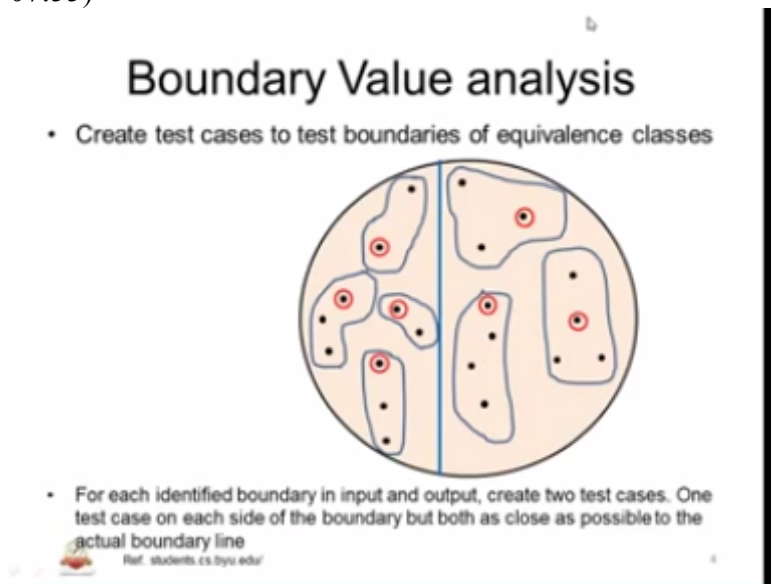
Today we will go through that black box testing of technique called boundary value of analysis,
(Refer Slide Time: 05:28)



So what is the boundary value of analysis? First we understood that the equivalence partitioning with the help of valid and invalid forms of, valid will have the humors number of this black box within the first five, and the second five have the invalid test cases, and the first level we are going to through in terms of, say this one set, this is another set so like this we have valid some for are there for invalid, equivalence of invalid 1, equivalence valid 2, equivalence valid 3, etc..
So only we have equivalence invalid, 1 we have equivalence invalid 2, EQ 3,EQ 4, so there are four valid equivalence classes 3 invalid equivalence classes, so out of this group of equivalence classes, we can select any one which is appropriate for that particular requirement, now again we might have one or more covering within the equivalence class.
It is very important for us to understand that typical requirement behavior also, because we did to see most of the requirements lying with the bounded values K and B so surrounding A and

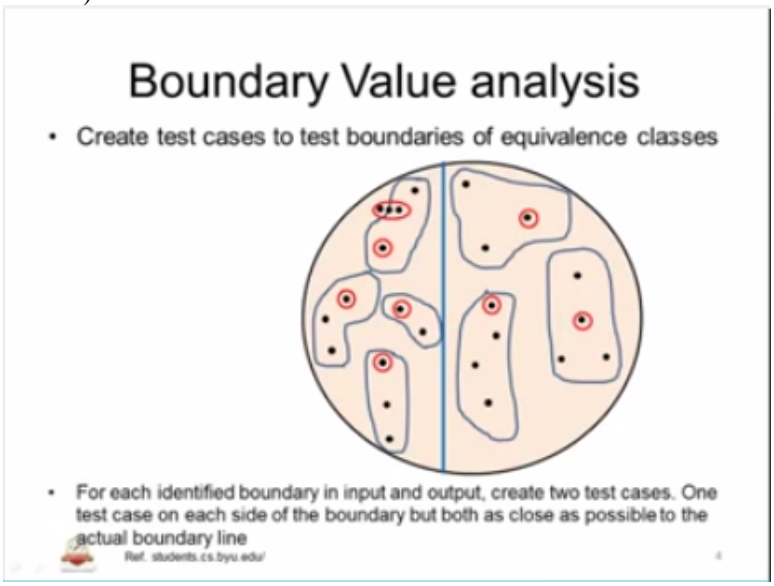
surrounding B how we are going to test it that also we need to conquer which is nothing but rebounded values which will replace the test, so the continuation of the same equivalence

(Refer Slide Time: 07:55)



Strgestic here we have the boundary value of analysis so we choose a selection that the things we have done of each group of equivalence classes are valid or invalid one test cases we have selected, we can see that round mark 1 as chosen taste cases, so in input condition has to specify a range bound by the entries, the sequence design should be divided value A and B and also above A just below B suggest, like this we are going to have it

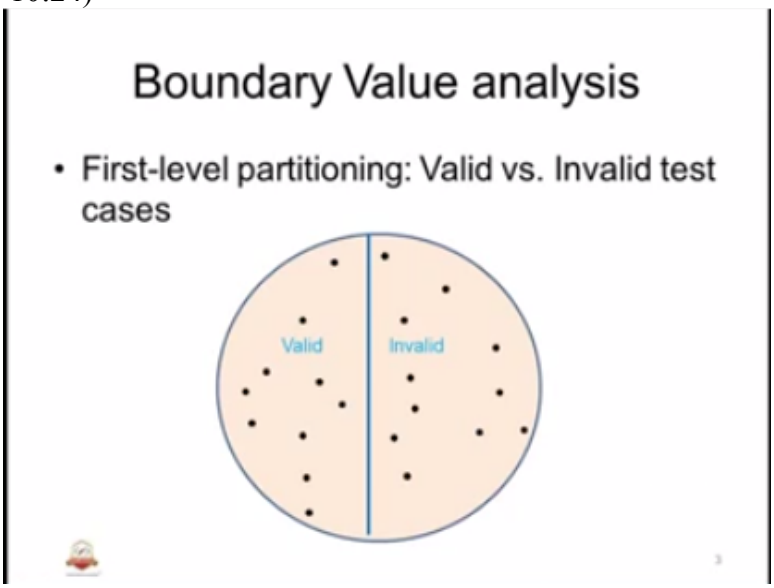
(Refer Slide Time: 08:53)



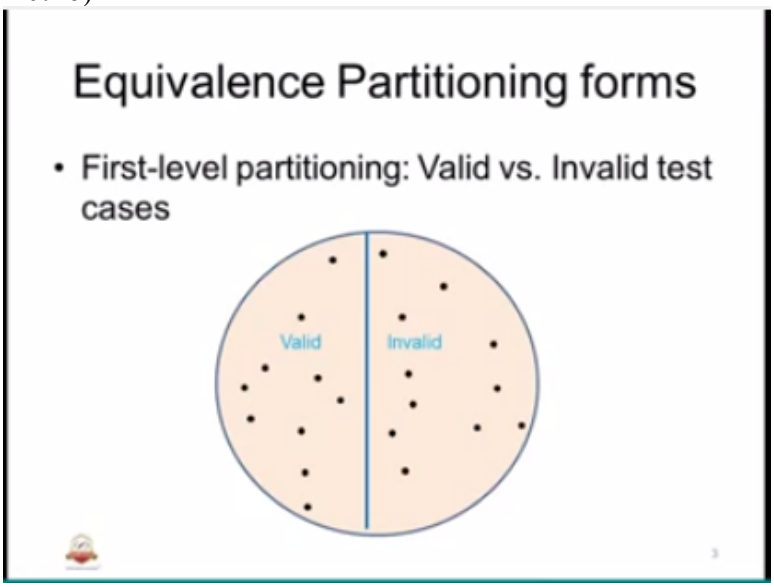
You, can see there is a boundary of A, just above A, just above B, suppose this slot is lying down on the boundary we have to test with the values which is above A, and below A, so this are first equivalence class similarly our equivalence class take it to have it, for invalid class also we need to have it in a same way, so which is nothing but the boundary analysis, so the first level is used to create a taste cases.

To test boundaries of equivalence class, for each identify boundary input and output create to taste cases, we know that two dots are here right, for the boundary conditions, two test cases are going to be there, one test case on each side of the boundary but both has close responsible to the actual boundary there, that means how the system behavior is for that the boundary inputs one for the low, one for the high or it could be one for the negative and other one could be for positive or one for the below or one for the above, so likewise they are going to have boundary values defined, so I will repeat again

(Refer Slide Time: 10:24)

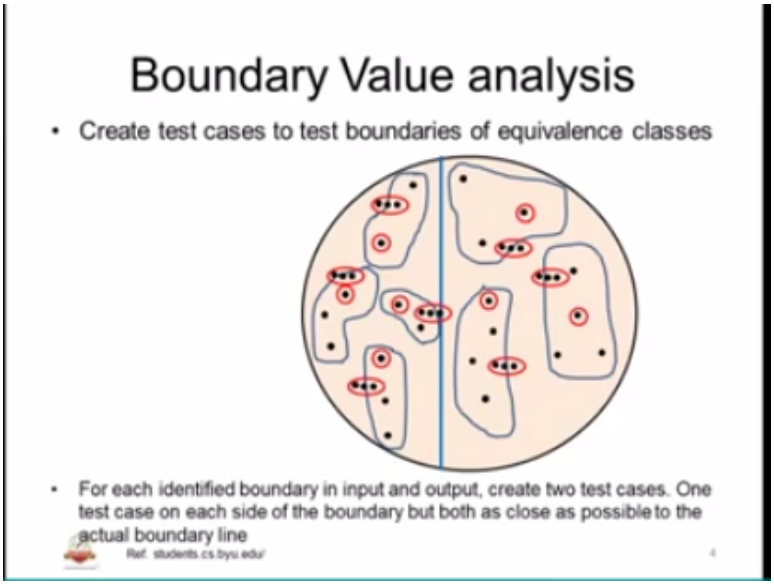


(Refer Slide Time: 10:26)



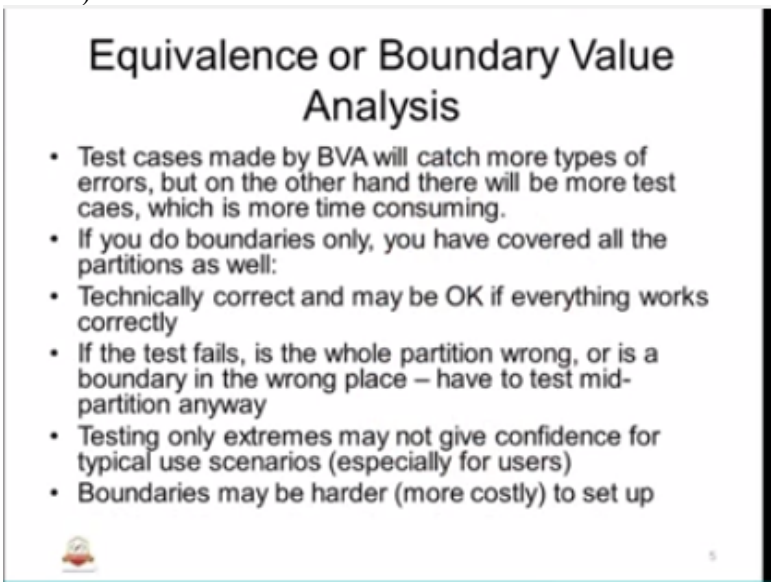
We have valid and invalid classes okay,

(Refer Slide Time: 10:29)



We are going to define them as equivalence classes valid and invalid, we have invalid criteria looking that each one good test section we are going to have it, the valid and invalid also we have that whether we are going to had for that, the requirement line with the bounded values which help of boundary inputs and outputs, so we can see the each of the equivalence classes valid and invalid

(Refer Slide Time: 11:08)



We have a boundary condition defined okay; the test cases made by boundary analysis will catch more types of error you know why? Because the system behavior boundary could change suppose a system is to take the variable of 1.0 to 5.0 and we are trying to test .9 or we are trying to test 4.9 or 5.21, so system should be help that values, we should predict the next value case of catching the errors or issues at the boundary is or lectured but this advantage will have test cases on another hand there be more test cases, which is more time consume, definitely it time consume because for each of the boundary conditions but equivalently important to identify the boundary value and analysis

Test cases if you boundary so only have covered all the functions of parts means, definitely it could cover all the parts, but whether to have an intimate value along with the boundary conditions, but what is really strictly correct and view okay, if everything once is a correct, so better to best practice to have an intimate value along with the boundary, so that means you have 1 to 10 has requirement inputs is good to have a boundary analysis for 1 and 10 and intimate the values such as 5 so,

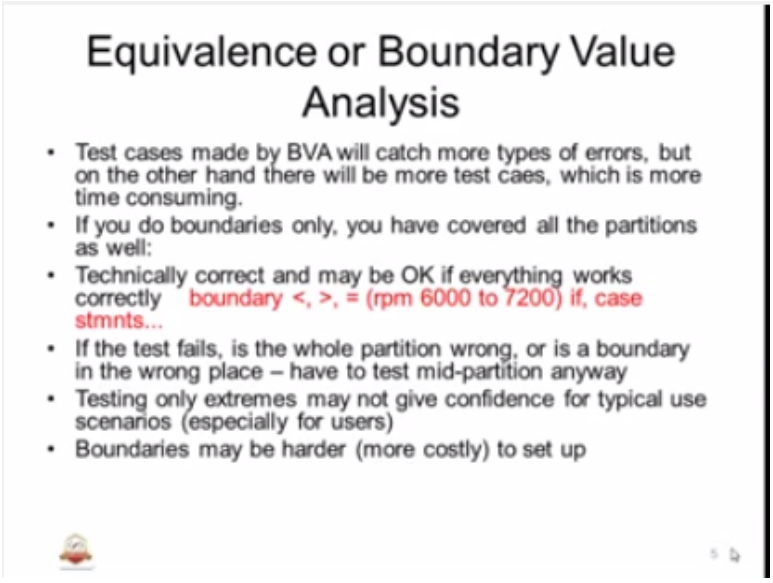
How many test cases we will have for 1 we have upper boundary and lower boundary, 2 test cases for the 10 we have upper boundary and lower boundary as 9 and 11 two more cases will become for the intimate value that is equal to 5_{th} one so probably 5 test cases for 1 to 10 if a test fails the whole partition is wrong, it is the not the test outcome or the test result, what we are trying do that partition if that test felts probably are in the boundary in the wrong place have to test mind partition anyway, testing only extremes may not give the confidence for typical uses it could have an intimate and good value of like 5,

when we consider range 1 in to 10, usually the system from 1 to 10 like 2 it could be 3 or 5, 6 or whatever it is, so better to identify one intimate value, good example is though the concept is 0 to 120, but there are less chances that drives in to 0 speed, so in a very few seconds it drives more than 150 to 200 kilo meters, so trying it or test once, the boundary condition is good but it is very important, that system is coverage table when it drives a range around in to 40 to 60 kilo meters, so that is what is called as normal testing cases, that really gives a good combination about the particular time, sometimes it is very difficult to spread the boundaries why? Because we know that cards Ceridian 0 to 50 but going by theory,

We may not able to test it right, so how you will go to fetch it? Are you going to take it reverse, it definitely not show as minus how are not sure, but we should have some stabilization are which is good to be coteries why? We need it is we need Spector in terms of theoretically were core mates of collapsed, if the speed is less than 0, or RPM it could be speed, or it could be the Indian RPM which goes behind in the certain level, oh! What is the behavior of that, likewise we need to have that testing, but to test that tricks may be challenging an instruction in terms of value, so we need to plan it properly, so basically boundary values requirement of the equivalence cases partitioning, instead of choosing any boundaries that equivalence cases, the main instruction was focus on the boundaries for each of the class, the idea is used to select one test case for each boundary of the equivalence class, the properties of what is test cases is belongs to find the equivalence class

It has value that, it is preferable on that means? One good value in to select out of this class are at least reasonably close to the boundary of particular equivalence class, so the main reason is boundaries are important is that, they are gently used by programmer, so the control also execute the program right, so programmers are implementer, the

(Refer Slide Time: 17:26)



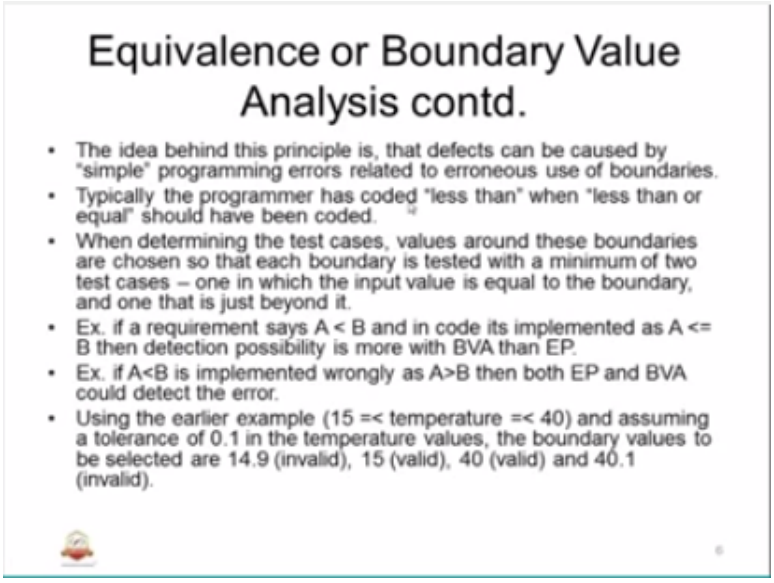
boundary values, as a boundary conditions <>= so if that is, RPM is 6000 to 7200, definitely he has consider the lowest value as the highest value while implementing the core, so that is why? We need to have a boundary value or analysis, so to control the execution, the core or the implement would have these control of equations for this values, definitely we have or implementing this.

If test case statements all this to be explained on boundary value analysis, so while doing this, it is bound to happen, that it could have implemented wrongly or it could have made some mistakes while implementing, so we will study that what are those implementation,

There are two different equivalence classes, the two sets of the boarder and then will be test for the boundary in both equivalence classes, the coverage is measured by dividing the number of executed.

So all these aspects we need to consider okay, in other terms we will detail about equivalence or boundary values the idea behind this principle that defects can be caused by simple code and errors related to erroneous or use of boundaries less than, greater than, typically the programmer has coded errors less than, when less than or equal should have been coded.

(Refer Slide Time: 19:25)



When determining the test cases, values around these boundaries are chosen, so that each boundary is tested with a minimum of two types of test cases, that means lower one and upper one, one which the input value is equal to the boundary and the one that is just beyond it, so this way we are going to design the requirement and the issues of the implementation, so example let us see if a requirement says A<B and in code its implemented as A<=B and then detection possibility is more or with BVA than E,BVA means boundary value analysis EP is equivalence partitioning.

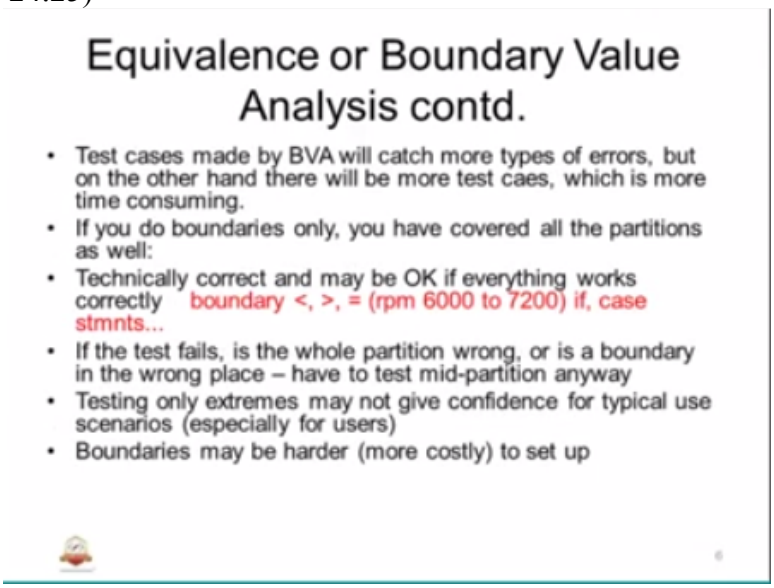
So this will bring out the issue of boundary values when that implemented are wrong again as A<B like as A<=B one more example as A<B is implemented wrongly as A>B also typically wrong and both EP and BVA could detect the errors, because of whole purpose of test is collapsed here, so definitely A of test is gone to be caught, the above one is detect the values equal to conditions of the boundaries to be caught in the BVA boundary value analysis, then a EP will not oriented that of the equivalence conditions, so going by the earlier example, in the previous session (15=< will less than or equal to temperature which is equal to less than =< 40) that means temperature level should be 50 <=40 temperature that means temperature level should be 50 as well as 40 were assuming a tolerance of 0.1 temperature values selected are 14.9, 15, 40 and 40.1 so these probably equivalence classes along with the boundary of cases so whatever goes invalid equivalence classes

Here we have 40.1 are invalid, 15 is normal and 40 is valid, because had equal not be there, and then we would had a valid classes that is 39. Suppose we go equal to, we start 15; we should have a 40 on valid code, similarly if there is no equal for related 40. And 39 could have valid equivalence code classes, 40 would have been an invalid equivalence classes and off course we have point 1 definitely in terms of probably we need to have the point 1 is consider as lower side of applying to the 15, we can have as well 15.1, here the 39.9 right, which know the valid and invalid equivalence classes and the boundary value analysis

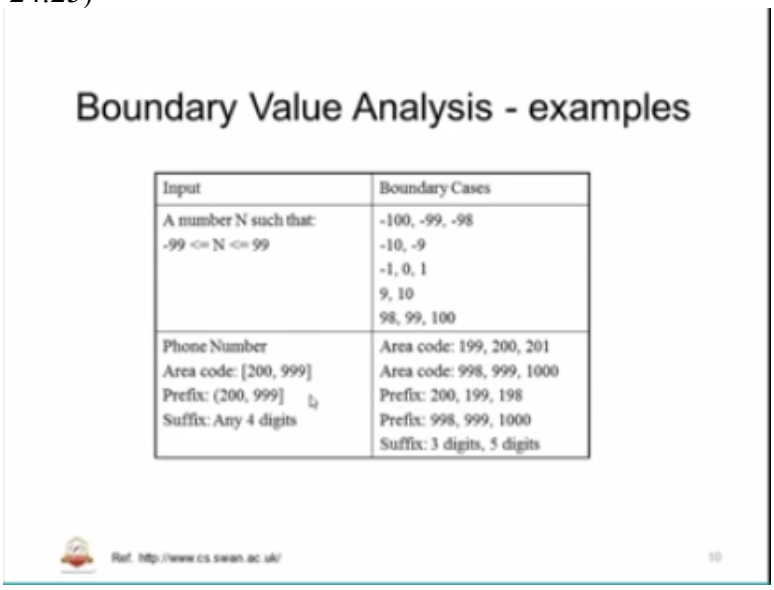
So these two techniques are very important tests boundary values are defined as equivalence classes partitioning we know, instead of choosing and each equivalence classes focus around the

boundaries, so these are the main things about this okay, so this about boundaries values for temperature

(Refer Slide Time: 24:25)



(Refer Slide Time: 24:25)

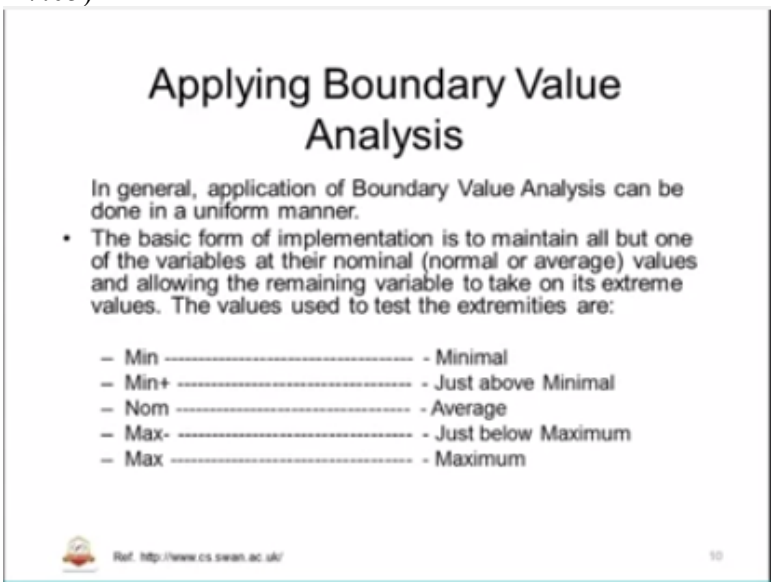


We look in to some more examples the frame example what we had to equal the partitioning N digit, +99 what are the boundary classes? Here no question of valid and invalid classes, so it is all will have it, but out of it which a word cases, we have to selected it as a valid or invalid equivalence class, so focus on boundary cases, so what is the boundary case ? < Or =N so here are the boundary cases, that we can have – 100 -99 -98,-99 as well is the boundary case, because on the executed term, H of the long boundary,

So we have -1 01 9 and 10, 98 99, 100. So these intimate value, because these all valid and equivalence classes of the range, so we need to have this along with the boundary cases, the next example is about phone number, so we will have a boundary cases, area code-199 100,201 at the lower boundary and the higher boundary 98, 99 and 1000,56, we know that 200,199,198 have the

boundary and higher boundary is 98,99, 1000 and we had some suffix any 4 digits, so we should have 3 digits as well as 3 suffix and 5 digits as another inputs of suffix so these are about valid equivalence classes with boundary conditions and it can allowed to suffix it, it is very important aspects of important value analysis okay, So we have gone through example

(Refer Slide Time: 27:05)



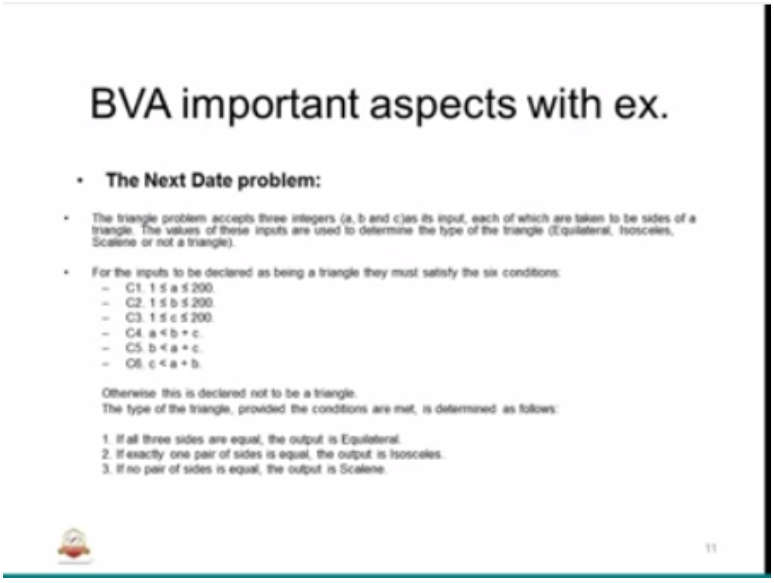
In general application of boundary value analysis can be done in a uniform manner that means, probably we can select based on the input what requirement says, suppose requirement is complicating having the numerous input and output then better to have a defined required table as well said to have a input table identifying all the combinations first identify the combinations then the next step is go for identifying the boundary value analysis and equivalence partitioning Equivalence.

Partitioning it is rectifying first, and then arrangement or had compliment partitioning conditions, so that is what should be the general behavior especially it is important for complex requirements, so we know that we are going to adjust, first requirements each requirements a how it can be tested? If that requirement beats the requirements to support, we may know that how the grouping can be done, that we have seen the in our, it need to suggest in a uniform manner.

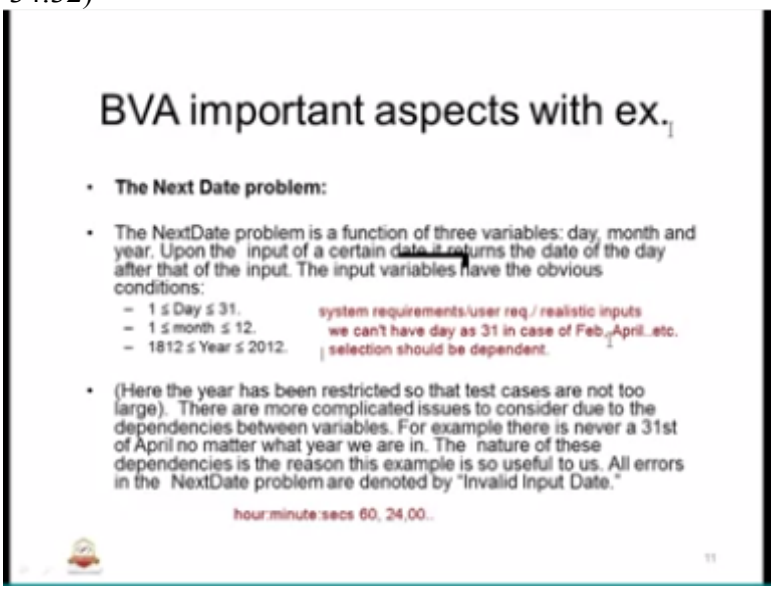
The basic form of implementation is to maintain all but, one of the variables at their, the normal values and allowing the remaining variables to taken on its extreme values, that means normal or average values first we have okay, den allowing the remaining variable to take as extreme values, the values is used to test the extreme case are below that means, that were we seen in the previous example, to privies on a we are going to have a one of complex rate requirements, and the values is used to test the extremities,

So we are going to have the minimal, and little more than that, above minimal and then nominal average or normal, and then we have a maximum value, then maximum value little reference there, below maximum, extremities values we are going to consider for the arriving it forming the boundary value analysis, this is very important for have it okay,

(Refer Slide Time: 30:10)



(Refer Slide Time: 34:32)



Boundary value analysis have an important aspects to focus on next date problem, next date problem is a function of three variables day, month, and year, upon the input on certain date, it returns the date of the day of the input, the input variables have the obvious conditions like below, day should be 1 to 31, 1 should be 1 to 12, year should be 18 to 2012, this is the definition of three variables, we know this all boundary, similarly we have a clock also, so what is the next date problem we going to have, here the year has been restricted, so that test cases are not too larger, why? Because we know that the system can take after 2012,

So there are more complicated issues to consider, due to the dependencies between variables, that means variables are very important to, so that dependencies should be consider, for an example there is never a 31st April it define the dependencies complicate based on the month right, if you have a February or if you have a month, so we are not going to have that 31 input right, so we cannot have test case trying to say you 31, for the month of April or the month of next date or given month, so no matter, what year we are in, day and month are depends, so the nature of this

dependencies is the reason, for an example is so useful to us, all errors in the next date problems are denoted by Invalid input date

So very important thing also we need it to understand that, we should be aware of the requirement or user requirement why? They important is, one value is depending on the other one, here the month is depending on the day, system which takes date are important, it is not what are UI, if you feed month as April, April cannot have a day as 31, so we cannot have a day as 31.

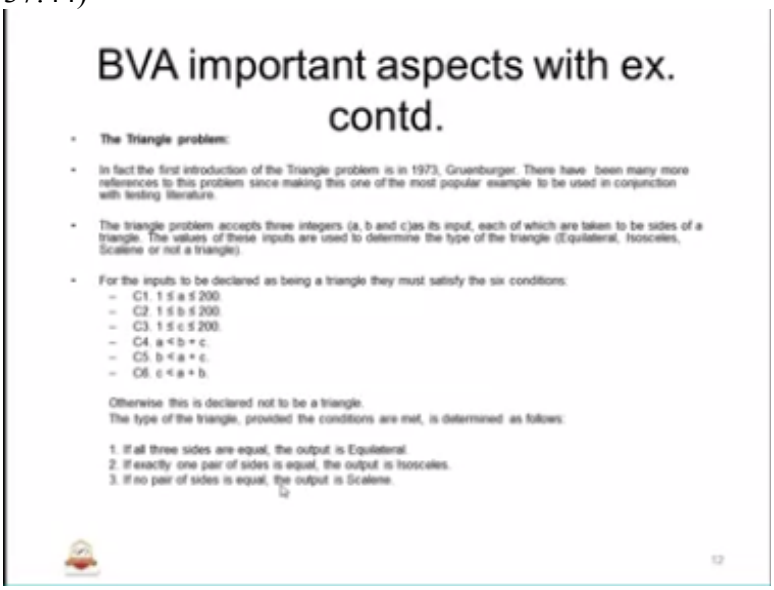
In case of February or April, etc... so, we need to have a selection in a such way that, selection should be depended, that means we need to understand, what are the component comes transfers that needs to be consider, so these things very important, not to just enough to have boundary values that because day feeds take 1 to 31, so also important to understand, these are the other aspects for dependencies in terms of the variables, similarly let us define a clock, so we have a hour, we need to select the seconds, so this again will have an dependencies in to the, we cannot go beyond 60, 60 it will go for next hour, similarly we go for 24 hour, we will have the follower of 0, 01 or 0.

Likewise we have ratio of a next date problem, so we need to have the, probably of the boundary values, it is important to have a understanding of restricted inputs, we are going to feed for the boundary value analysis, so we should not get stuck with the problem such as the next date problem, one more is there is called a triangle problem, I will just go through simplify.

So that these also can be consider, so interrupt the first introduction of the triangle problem in a 73 by equivalent on the, here have been MANU more reference problem, we just making one of the most popular example to be used in conditions with testing the literature

The triangle problem access the integers A B C, as an input example, each of which taken to the sides of a triangle, that means we have a three sides either or equilateral whatever you want to call? So this three inputs are sides of a triangle, the values of the inputs are used in the type of the triangles, so what type of a triangle? These three inputs are going to decide, so a triangle could be one of this like an equilateral,

(Refer slide Time: 37:44)



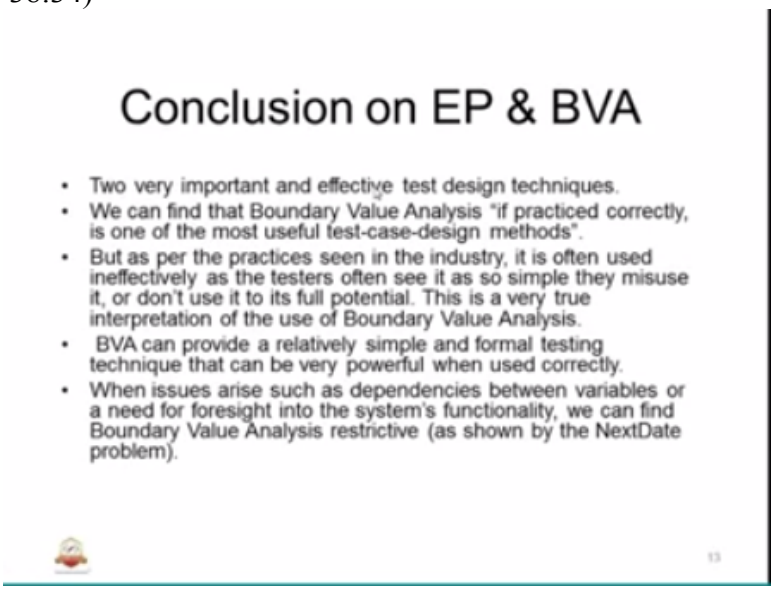
Basically it is not at all a triangle, so one of all this, these all the factor based on the A B C, for the inputs to be declared as a being a triangle these must satisfy the conditions, so C1, C2, C3, C4,C5,C6, so we have this conditions, well before we started as why? Because first we define it is being a triangle or not, so to define that, we need to have a triangle to define with conditions 1 as, A < =1=200, B<=1>=200, similarly we see 1 and 200, then we have other binding conditions, A should be >B+ A, and B> A+B, C>A+B, so this conditions basically in order to form the triangle, so otherwise never called at the triangle, so the type of a triangle provide the conditions are mate determine as follows

If all three sides of equal the output is equilateral, so exactly one pair of sides is equal and output side is raises, no pair of sides is equal, output is raises, no pair of sides is equal, and output is remaining, to define the triangle, we are going to have what type of triangle, so we need to be very careful in the choosing the test case.

Also it should be realistic in terms of the inputs, first of all define the criteria requirement has been laid out, first we need to understand that is what am always the test case has to have the good knowledge of system under the test

We should not get stuck within the testing aspects or not good in terms of system selecting techniques of strategist is not good, first we should understand the 5th term we should design the test case, a test is when behave in terms of executing and producing the output of the under grade embedding the system okay,

(Refer Slide Time: 38:34)



So to conclude on the equivalence partitioning boundary analysis, where some important points that we need to see is, these two are very important and effective test design techniques, we have to have actually equivalence partitioning boundary analysis, we can find that boundary value analysis, if practices correct, actually is one of the most useful for test design, test case design tech methods.

so boundary value analysis have to be correctly practiced or implemented having consider on all the issues and considering all the ranges, consider with different examples that we have seen and

effectiveness of boundary conditions etc…So if it is correctly practice, it will be very useful in terms of finding.

Under the embedded software, but as per the practices save in the test drive, it is off on use the in effectively, as the testers often see it is so simple, they misuse it. I believe that it is going to work, because when I plug it I apply a power, it is behaving good, and it will changes actually is good, as a black box, I feel that system is good, and the user convex, that is not a case

We should not consider that has the primary input, or we should think in terms of having a plug and going to challenge it, in terms of in all the issues, or the tester would not have used it potential.

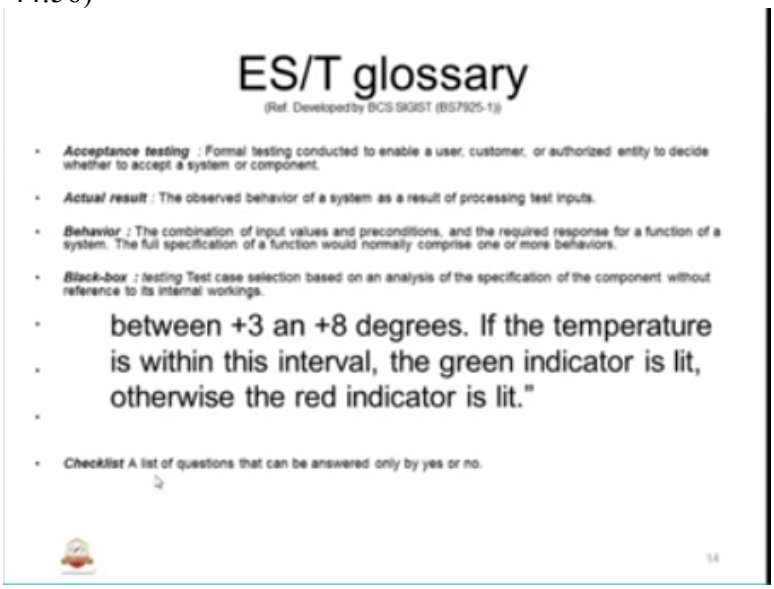
If you get the mobile or handset of or any instruments, what basically is do as a user he will plug it, he will keep on, he will try to dial it, so as a normal behavior, so he gets the concern that is fine, but instead of using its full potential, definitely he has to think out of the box in terms of testing it effectively, so far testing it is effectively, he needs to understand what is capable of, and now what is the specification, what are the conditions at it can observe, what are the techniques he can apply? Effectively it is important

So that, will be to bring out all the test effects, which are the having issues in terms of implemented, and this more important why? Because boundary value analysis, will bring the effective way of bringing out the works, so BVA can provide a relatively simple and formal testing that can be very powerful, when used correctly, that means we start formally with that technique of identifying of equivalent classes, and identifying the boundaries, the inputs that requirement are identify then selects the test case based on the inputs, and the criteria, apply the test, and find the errors, when issues are arrives such as dependencies between variables or need of, foresight in to the system functionality, we can find boundary value analysis restrictive example we can take it as next date problem,

realistic boundary values should allow for the user to give month of April day as 31, this is not a realistic input for an embedded system having a day and incremented functionality, so we cannot accepted it, so that is the restriction for boundary value analysis means, we should take a call in terms of boundary value analysis, taking out the such cases, for such issues. So that is the conclusion of equivalent partitioning and boundary value analysis

(Refer Slide Time: 42:42)

(Refer Slide Time: 44:56)



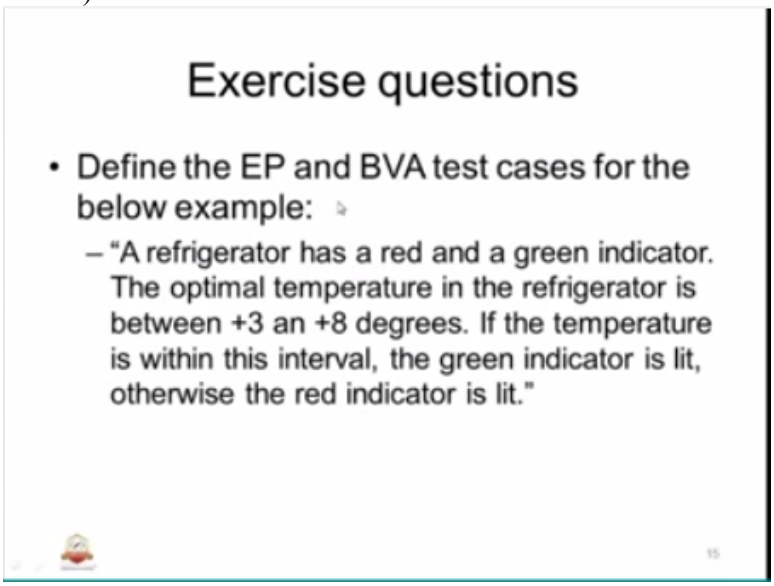
We will have a some of the glossary in an embedded software testing, and in to each divided in to different sections, so we will just go through that, aspect testing, we know that, this is a formal testing from these aspects, actual result, the result behave an actual system as glossary in protects, you know what is behavior?

The combination of input values are the conditions and required response function of the system, so that is the behavior, if specification function would normally comprise one or more behaviors, you know what is a black box, testing a test case selection based on the analysis of specification of the component, with reference to its manner workings, or internal implementations details with the black box.

And white box we know about the logic and program for, it is an alphabetic order, that is why, it is not tested all, so boundary values and input value and output value which is on the boundary between equivalence classes our incremental resistance either side of the boundary, could be upper, below high, or low etc..

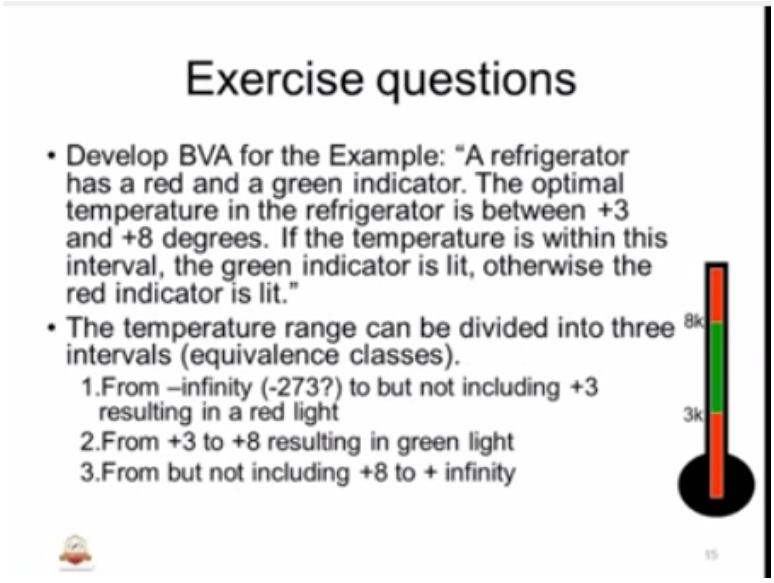
So boundary value analysis is in the technique as equivalence class, for an example component which is used to design input represent, of boundary values, application of conforming that, the system or components complains the specified requirements, and acceptable for operation is used, so we will talk about the application, checklist a list of questions that can be answered, basically is used by support of people for doing the values and rectifications To be done as a tollgate before the product is tested and released, mostly it done by the departmental test BVA. Okay,

(Refer Slide Time: 44:57)



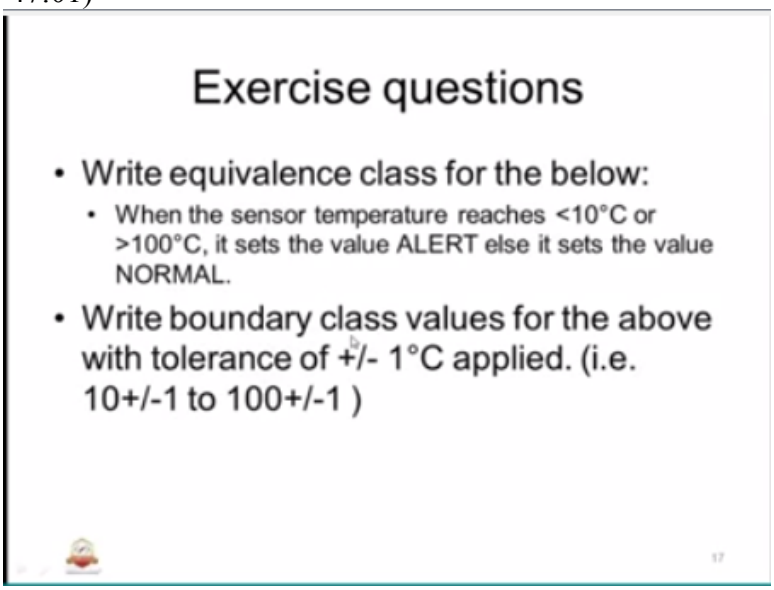
we have an exercise question, define the EP and BVA test cases for the below entries the below example says, a refrigerator has a red and green indicator, the optimal temperature in the refrigerator is between 3 and 8 degrees, if the temperature is within this through all, green indicator is lit, otherwise the red indicator is lit, that means the indicator is in the refrigerator will indicate a green is temperature is between 3 and 8 is lit, if it is beyond that will show it as red, so we need to draw a equivalence partitioning, as well as boundary value analysis specifications for the below

(Refer Slide Time: 45:54)



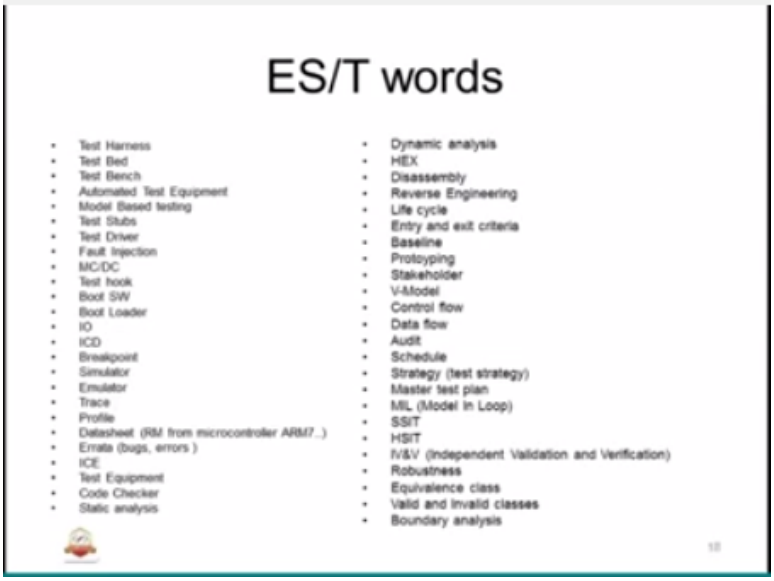
One, more x ray, I think it is an continuation of the previous x rays, we can say the diagram of the temperature indicator, develop BVA for the example A refrigerator is the optimal temperature 3 and 8, if temperature is interval and the green indicator is lit, the temperature range can be divided in to three intervals, so I will give few inputs, accordingly you can define the boundary value analysis, and the equivalence partitioning, so from infinity it could be – any value but not including 3, which is result in red, here down we can see 3 and below or red and option above a red, between 3 and 8, the temperature get indicated, from 3 to 8 is green, but not including 8 to the higher values is red

(Refer Slide Time: 47:01)

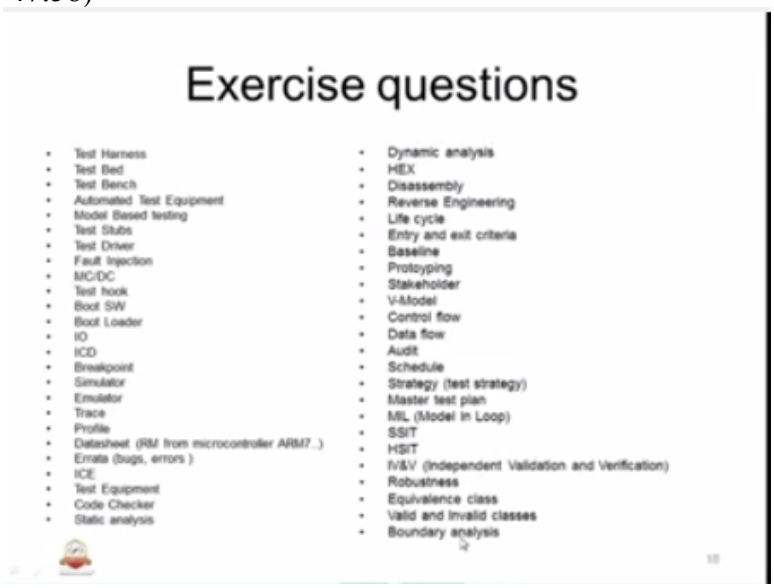


So One more x rays I think x rays is given in the equivalence partitioning will give an extenuations for this, for rate this equivalence class of the below values, the sensor temperature reaches <10 degree Celsius or 100 degree Celsius, it sets the value normal, so the rate boundary class value is for the above with the tolerance of +/- 1 degree Celsius that means less than 10 +/- 1 to 100+/-1) should be applied for doing the boundary values analysis okay,

(Refer Slide Time: 47:40)



(Refer Slide Time: 47:58)



So some of the embedded system, we will go through, we have defined all this previous section, we are going to have a equivalence class, and boundary analysis, in new words have done today we will see next date problem, and nominal average normal we will had it okay, so will had a nominal or normal average, it is in type of inputs which is set to the system, so that is about the boundary value analysis that is very important to learn boundary value analysis and equivalence partitioning, because the entire design is founded to techniques for the embedded software testing . So that is about boundary value analysis, so it is very important to learn boundary value analysis partitioning is founded with two techniques for the embedded software techniques.