CMR INSTITUTE OF TECHNOLOGY

Internal Assessment Test 1 – March. 2018

Scheme and Solution

Sub:	Software Testing							**Code:**	15IS63
Date**:**	13/03/2018	Duration:	90 mins	Max Marks:	50	**Sem:**	VI	**Branch:**	ISE

Note: Answer any five questions:

1 a) Define the following: i) Error ii) fault iii) failure iv) incident v) test vi) test case (1 M each) — 6 M

Error: People make errors. A good synonym is "mistake". When people make mistakes while coding, we call these mistakes "bugs". Errors tend to propagate; a requirements error may be magnified during design, and amplified still more during coding

Fault : A fault is the result of an error. It is more precise to say that a fault is the representation of an error, where representation is the mode of expression, such as narrative text, dataflow diagrams, hierarchy charts, source code, and so on. "Defect" is a good synonym for fault; so is "bug". Faults can be elusive.

Failure: A failure occurs when a fault executes. Two subtleties arise here: one is that failures only occur in an executable representation, which is usually taken to be source code, or more precisely, loaded object code. The second subtlety is that this definition relates failures only to faults of commission

Incident: When a failure occurs, it may or may not be readily apparent to the user (or customer or tester). An incident is the symptom(s) associated with a failure that alerts the user to the occurrence of f**a**ilure.

Test : Testing is obviously concerned with errors, faults, failures, and incidents. A test is the act of exercising software with test cases. There are two distinct goals of a test: either to find failures, or to demonstrate correct execution.

Test Case: A test case has an identity, and is associated with a program behavior. A test case also has a set of inputs, a list of expected outputs.

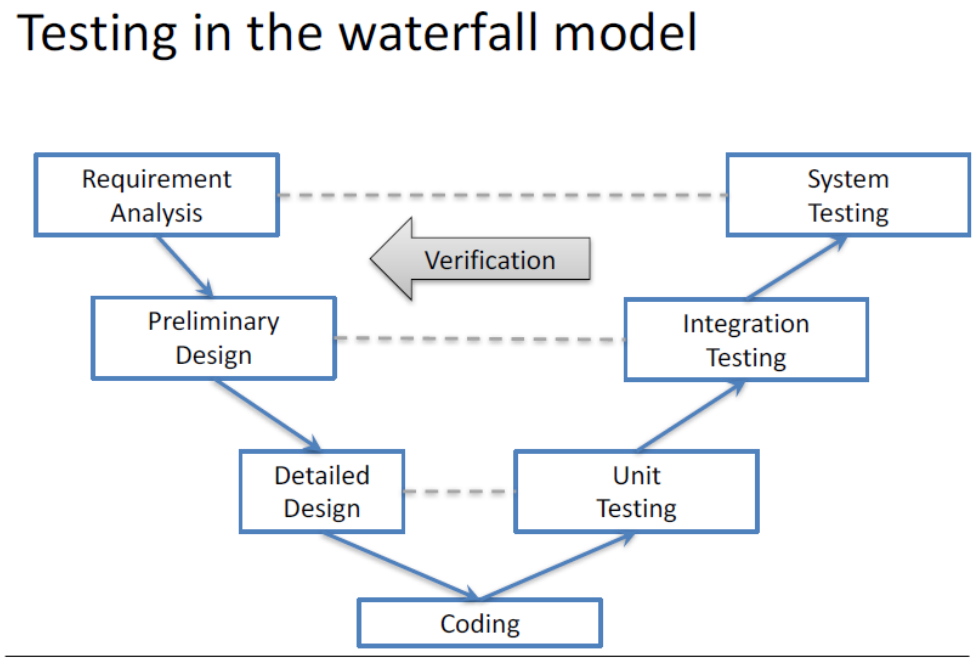
b) Explain the levels of abstraction and testing in the waterfall model

LEVELS OF TESTING

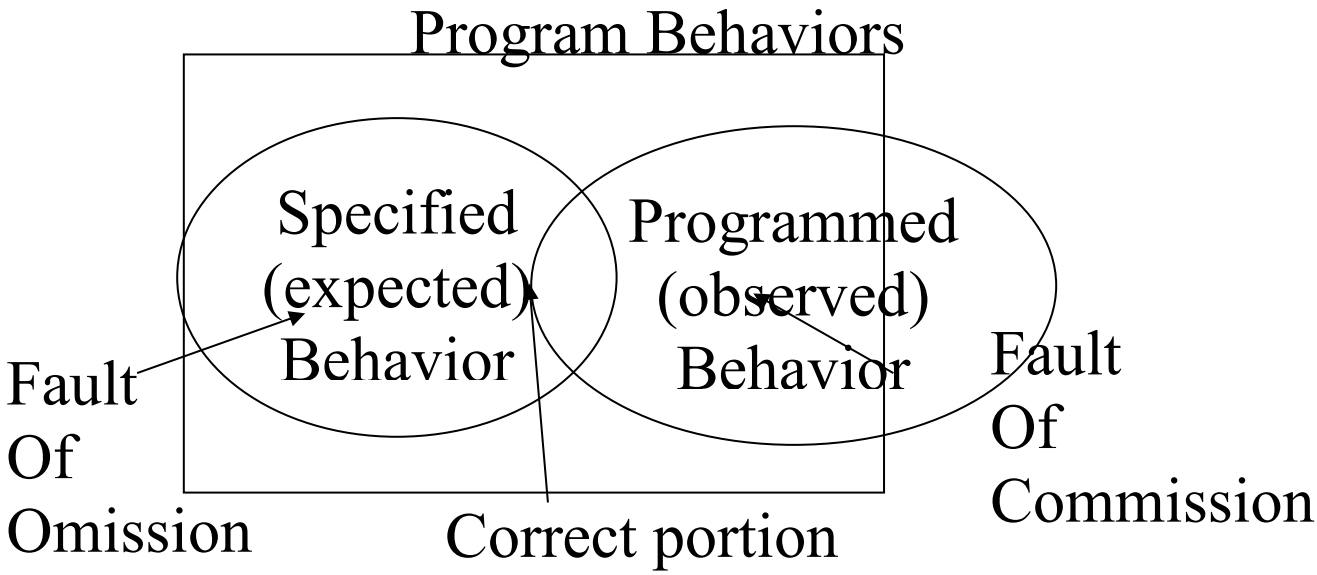
Levels of testing echo the levels of abstraction found in the Waterfall Model of the software development life cycle. While this model has its drawbacks, it is useful for testing as a means of identifying distinct levels of testing, and for clarifying the objectives that pertain to each level.

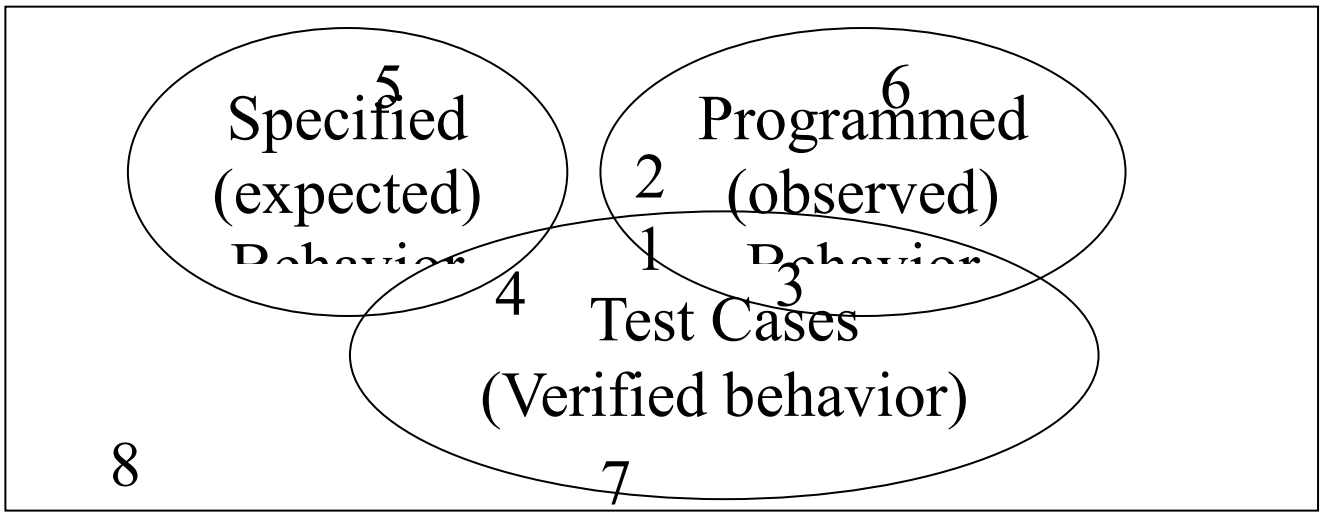
To enhance the quality of software testing, and to produce a more unified testing methodology applicable across several projects, the testing process could be abstracted to different levels. This classification into different levels introduces some parallelism in the testing process as multiple tests could be performed simultaneously. Although these levels are best suited for the waterfall model, (since the levels correspond directly to the different stages in the waterfall model) the level of abstraction still poses to be useful across other software development models.

- Unit testing
- Integration testing
- System testing



2	Explain: i. Currency converter ii. Saturn wind shield wiper controller. Currency Converter description 5 M Saturn wind shield wiper controller description 5 M	10M
3	Briefly explain testing using Venn Diagram. (5 M) Testing is fundamentally concerned with behavior; and behavior is orthogonal to the structural view common to software (and system) developers. A quick differentiation is that the structural view focuses on "what it is" and the behavioral view considers "what it does".	10 M





- 2, 5: Specified behavior that are not tested
- 1, 4: Specified behavior that are tested
- 3, 7: Test cases corresponding to unspecified behavior
- 2, 6:Programmed behavior that are not tested
- 1, 3:Programmed behavior that are tested
- 4, 7:Test cases corresponding to un-programmed behaviors

If there are specified behaviors for which there are no test cases, the testing is incomplete. If there are test cases that correspond to unspecified behaviors Either such test cases are unwarranted, or
Specification is deficient: It also implies that testers should participate in specification and design reviews

Differentiate between functional testing and structural testing. (5M)

Functional testing is based on the view that any program can be considered to be a function that maps values from its input domain to values in its output range. This leads to the term black box testing in which the content (implementation) of a black box is not known, and the function of the black box is understood completely in terms of its inputs and outputs.

With the functional approach to test case identification, the only information that is used is the specification of the software. There are two distinct advantages to functional test cases: they are independent of how the software is implemented, so if the implementation changes, the test cases are still useful, and test case development can occur in parallel with the implementation, thereby reducing overall project development interval. On the negative side functional test cases frequently suffer from two problems :

1. Redundancies
2. Gaps in tested software.

Structural Testing

Structural testing is the other fundamental approach to test case identification. To contrast it with Functional Testing, it is sometimes called White Box (or even Clear Box) Testing. The clear box metaphor is probably more appropriate, because the essential difference is that the implementation (of the Black Box) is known and used to identify test cases.

4 What is boundary value analysis? Write the test cases using boundary value analysis testing for triangle problem explain robustness testing and worst case testing. 10 M

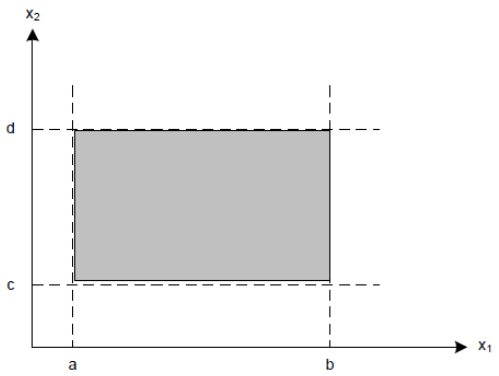
Explanation of BVA with test cases of triangle problem: 6 M
Explanation of robustness testing and worst case testing: 4 M

Boundary Value Testing

• Two considerations apply to boundary value testing
– are invalid values an issue?
– can we make the "single fault assumption" of reliability theory?

• Consequences...
– invalid values require the robust choice
– multiplicity of faults requires worst case testing
• Taken together, these yield four variations
– Normal boundary value testing
– Robust boundary value testing
– Worst case boundary value testing
– Robust worst case boundary value testing

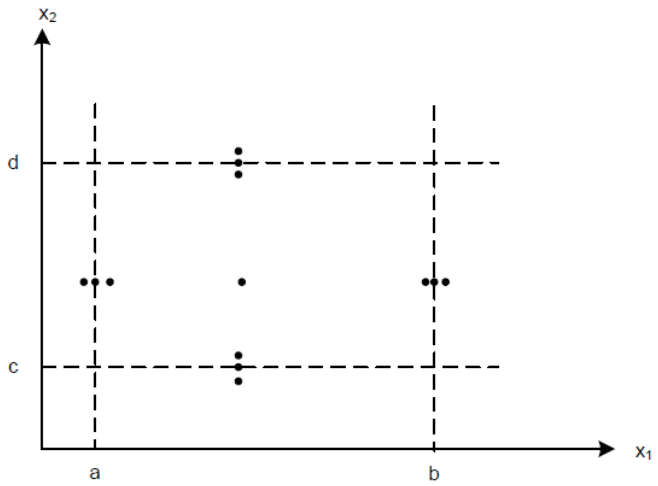
Input Domain of $F(x_1, x_2)$
where $a \le x_1 \le b$ and $c \le x_2 \le d$



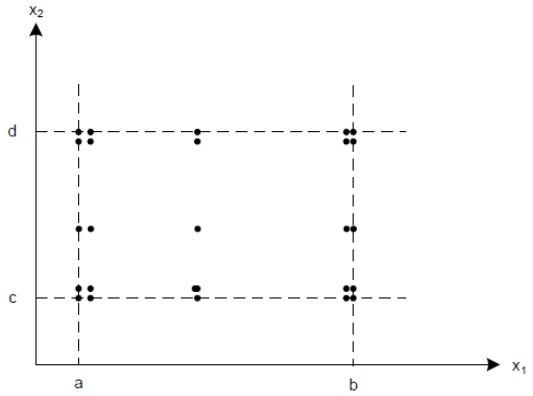
5 M

Robustness Testing

Stress boundaries
• Possible advantages
– find hidden functionality
– leads to exploratory testing



Normal Worst Case Boundary Value Test Cases



Responding to the single-fault assumption.

5 a) Explain Test and Debug Cycle with a neat diagram 10M

Diagram – 2M
Explanation – 4M

b) List out the quality attributes and explain in brief. (4M)

6 Briefly explain weak normal, strong normal, weak robust and strong robust equivalence class testing with an example. 10M

weak normal, strong normal, weak robust and strong robust equivalence class testing (8 M)
Example (2 M)

Weak & Normal Equivalence Class Testing
- A function F, of two variables x1 and x2
- x1 and x2 have the following boundaries and intervals within boundaries:

$a \leq x1 \leq d$, with intervals [a, b) [b, c), [c, d)
$e \leq x2 \leq g$, with intervals [e, f) [f, g)
- [= closed interval, (= open interval

Weak Normal Equivalence Class Testing
- One variable from each equivalence class
- Values identified in systematic way