MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17624 **Subject Name: Software Testing**

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in themodel answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills).
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgment on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

1. [A] Attempt any THREE of the following: **Marks 12**

(a) What is entry and exit criteria of software testing?
(Entry criteria-2Marks, Exit criteria-2Marks)

Ans: **When to Start and Stop Testing of Software (Entry and Exit Criteria)**
Process model is a way to represent any given phase of software development that prevent and minimize the delay between defect injection and defect detection/correction.

- **Entry criteria** ,specifies when that phase can be started also included the inputs for the phase.

Tasks or steps that need to be carried out in that phase, along with measurements that characterize the tasks.

Verification**,** which specifies methods of checking that tasks have been carried out correctly.

Clear entry criteria make sure that a given phase does not start prematurely.

The verification for each phase helps to prevent defects. At least defects can be minimized.

- **Exit criteria**, which stipulate the conditions under which one can consider the phases as done and included are the outputs for the phase.

Exit criteria may include:

- All test plans have been run
- All requirements coverage has been achieved.
- All severe bugs are resolved.

(b) Describe Inspection under static testing.
(What is inspection-1Mark, Roles & explanation -3 Marks)

Ans: Static testing is to review the code without executing it. Formal Inspection is the most formal method in static testing. This method can detect all faults, violations and other side effects.
In this:
1. Thorough preparation is required before an inspection/review
2. Enlisting multiple diverse views.
3. Assigning specific roles to the multiple participants
4. Going sequentially through the code in a structured manner.

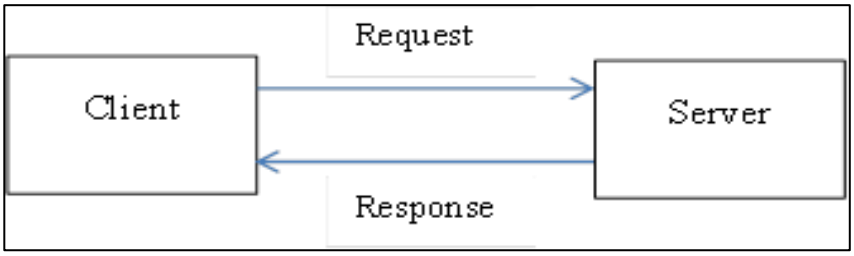
There are four roles in this.

1.Author of the code: the person who had written the code
2.Moderator: who is expected to formally run the inspection according to the process.
3.Inspectors: are the people who actually provide review comments for the code.
4.Scribe: who takes detail notes during the inspection meeting and circulates them to the inspection team after the meeting.

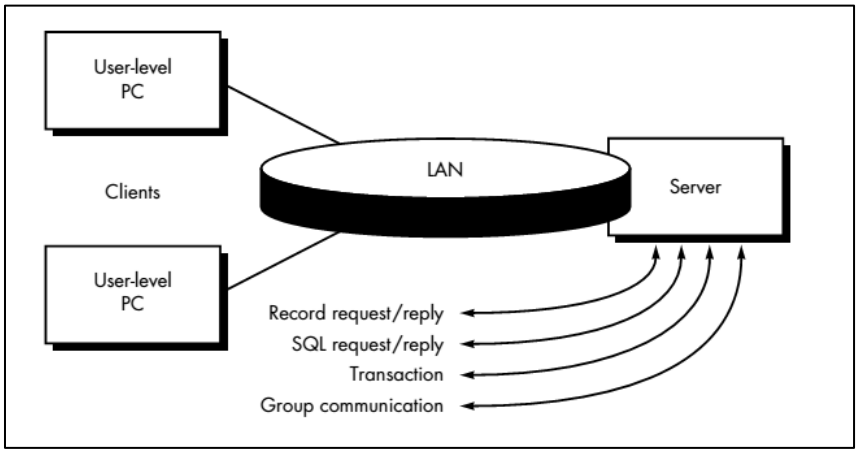
The author or moderator selects review team. The inspection team assembles at the agreed time for inspection meeting. The moderator takes the team sequentially through the program code. If any defect is found they will classify it as minor or major. A scribe documents the defects. For major defects the review team meets again to check whether the bugs are resolved or not.

(c) With the help of diagram describe client-server testing.
(Diagram-1Mark, Explanation-3Marks)
[Note: Any other relevant diagram and tests also can be considered]

Ans:



OR



In Client-server testing there are several clients communicating with the server.
1. Multiple users can access the system at a time and they can communicate with the server.
2. Configuration of client is known to the server with certainty.
3. Client and server are connected by real connection.
Testing approaches of client server system:

1. Component Testing: One need to define the approach and test plan for testing client and server individually. When server is tested there is need of a client simulator, where as testing client a server simulator, and to test network both simulators are used at a time.

2. Integration testing: After successful testing of server, client and network, they are brought together to form system testing.

3. Performance testing: System performance is tested when number of clients are communicating with server at a time. Volume testing and stress testing may be used for testing, to test under maximum load as well as normal load expected. Various interactions may be used for stress testing.

4. **Concurrency Testing:** It is very important testing for client-server architecture. It may be possible that multiple users may be accessing same record at a time, and concurrency testing is required to understand the behavior of a system in this situation.

5. Disaster Recovery/ Business continuity testing: When the client server are communicating with each other , there exit a possibility of breaking of the communication due to various reasons or failure of either client or server or link connecting them. The requirement specifications must describe the possible expectations in case of any failure.

6. Testing for extended periods: In case of client server applications generally server is never shutdown unless there is some agreed Service Level Agreement
(SLA) where server may be shut down for maintenance. It may be expected that server is running 24X7 for extended period. One needs to conduct testing over an extended period to understand if service level of network and server deteriorates over time due to some reasons like memory leakage.

7. Compatibility Testing: Client server may be put in different environments when the users are using them in production. Servers may be in different hardware, software, or operating system environment than the recommended. Other testing such as security testing and compliance testing may be involved if needed, as per testing and type of system.

(d) What is test plan? List test planning activities.
(Test Plan-1Mark, Listing Activities-3Marks)

Ans: Like any project, the testing also should be driven by a plan. The test plan acts as the anchor for the execution, tracking and reporting of the entire testing project. Activities of test plan:

1.Scope Management: Deciding what features to be tested and not to be tested.
2.Deciding Test approach /strategy: Which type of testing shall be done like configuration, integration, localization etc.
3.Setting up criteria for testing: There must be clear entry and exit criteria for different phases of testing. The test strategies for the various features and combinations determined how these features and combinations would be tested.
4.Identifying responsibilities, staffing and training needs

5.Identifying resource requirements
6.Identifying test deliverables
7.Testing tasks: size and effort estimation

[B] Attempt any ONE: **Marks 6**

(a) Explain GUI testing with suitable example.
(Explanation-4 Marks, Example – 2 Marks)

Ans: Graphics User Interface Testing (GUI) Testing is important part of application along with functionality as it affects the usability.
Example: Consider any website like MSBTE, google, yahoo or any login form or GUI of any application to be tested.

It includes following:

- All colors used for background, control colors, and font color have a major impact on users. Wrong color combinations and bright colors may increase fatigue of users.
- All words, Fonts, Alignments, scrolling pages up and down, navigations for different hyperlinks and pages, scrolling reduce usability.
- Error messages and information given to users must be usable to the user. Reports and outputs produced either on screen or printed should be readable. Also paper size on printer, font, size of screen should be consider.
- Screen layout in terms of number of instructions to users, number of controls and number of pages are defined in low level design. More controls on single page and more pages reduce usability.
- Types of control on a single page are very useful considering usability.
- Number of images on page or moving parts on screen may affect performance. These are high-priority defects. It has direct relationships with usability testing, look, and feels of an application. It affects emotions of users and can improve acceptability of an application.

Advantages of GUI Testing:

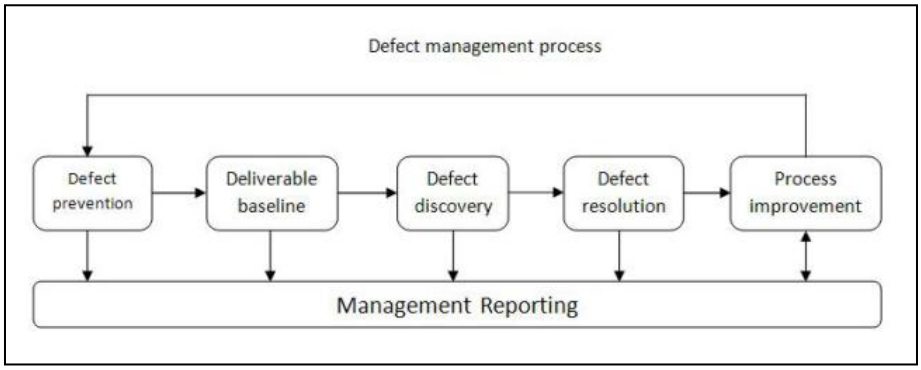
- Good GUI improves feel and look of the application; it psychologically accepts the application by the user.
- GUI represents a presentation layer of an application. Good GUI helps an application due to better experience of the users.
- Consistency of the screen layouts and designs improves usability of an application.

Disadvantages of GUI Testing:

- When number of pages is large and number of controls in a single page is huge.
- Special application testing like those made for blind people or kids below age of five may need special training for testers.

(b) Describe steps in Defect Management Process.
(Diagram-2Marks, Explanation-4Marks)

Ans:



1. Defect Prevention: Implementation of techniques ,methodology and standard processes to reduce the risk of defects.
2. Deliverable Baseline: Deliverables are considered to be ready for further development.i.e. the deliverables meet exit criteria.
3.Defect Discovery: To find the defect through the process of verification and validation.
4.Defect Resolution: Defect is corrected or corrective action is taken and notification is given to tester.
5.Process Improvement: To identify ways to improve the process to prevent further future occurrences of similar defects. i.e. Corrective and preventive action is taken for processes improvement.
Management Reporting: Reporting is about status of application and processes.

2. Attempt any FOUR: Marks 16

(a) What is Black Box testing? List any four techniques of Black Box testing.
(Black box testing-2Marks,Listing any 4 techniques -2Marks)

Ans: Black Box testing involves looking at the specifications and does not require examining the code of the program. It is done from customer's viewpoint. The testers know the input and expected output. They will check whether with given input they are getting expected output or not.

Different techniques of Black Box test are:
1. Requirement base testing
2. Positive negative testing
3. Boundary value analysis
4. Decision tables
5. Equivalence partitioning
6. State based testing
7. Compatibility testing
8. User documentation testing
9. Domain testing

(b) How to select a testing tool? Explain in detail.
(Explanation – 4 Marks)

[Note: Criteria or guidelines can be considered as an answer]

Ans: Criteria for Selecting Test Tools:
The Categories for selecting Test Tools are,
1. Meeting requirements;
2. Technology expectations;
3. Training/skills;
4. Management aspects.

1. **Meeting requirements-**

There are plenty of tools available in the market but rarely do they meet allthe requirements of a given product or a given organization. Evaluating different tools for different requirements involve significant effort, money, and time. Given of the plethora of choice available, huge delay is involved in selecting and implementing test tools.

2. **Technology expectations-**

Test tools in general may not allow test developers to extends/modify the functionality of the framework. So extending the functionality requires going back to the tool vendor and involves additional cost and effort. A good number of test tools require their libraries to be linked with product binaries.

3. **Training/skills-**

While test tools require plenty of training, very few vendors provide the training to the required level. Organization level training is needed to deploy the test tools, as the user of the test suite are not only the test team but also the development team and other areas like configuration management.

4. **Management aspects-**

A test tool increases the system requirement and requires the hardware and software to be upgraded. This increases the cost of the already- expensive test tool.

OR

Guidelines for selecting a tool:

1. The tool must match its intended use. Wrong selection of a tool can lead to problems like lower efficiency and effectiveness of testing may be lost.
2. Different phases of a life cycle have different quality-factor requirements. Tools required at each stage may differ significantly.
3. Matching a tool with the skills of testers is also essential. If the testers do not have proper training and skill then they may not be able to work effectively.
4. Select affordable tools. Cost and benefits of various tools must be compared before making final decision.
5. Backdoor entry of tools must be prevented. Unauthorized entry results into failure of tool and creates a negative environment for new tool introduction.

(c) Prepare six test cases for Admission form for college admission.
(Test cases -4 Marks)

[Note: Any other relevant test cases also can be considered]

Ans: Consider the college admission form having different fields such as Student's Name, Father's Name, Address,Phone,Caste,admission type,S.S.C percentage, SC Board, Submit button, Reset button.

Test Case Id	Test case Objectives	Input Data	Expected Result	Actual Result	Status
TC1	Name field	Any name(abcd xyz)	It should accept the name	The name is accepted	Pass
TC2	Phone Field	Any number having less than 10 digits(1234)	It should not accept. Should give error message "Please enter valid phone number"	Error message "Please enter valid phone number"	Pass

TC3	Phone Field	Any alphabets(abcde)	It should give error message as "Only Numbers"	Error message as "Only Numbers"	Pass
TC4	SSC Percentage Field	65	It should accept	It accepted	Pass
TC5	SSC Percentage Field	30	It should not accept. Should give error message.	Gives error message	Pass
TC6	Address field	Any characters(A-51, Market road, Mumbai)	It should accept.	It accepted	Pass

(d) Enlist any four benefits of automation testing.
(Any four benefits, Each benefit -1 Mark)

Ans:

1. **Save Time /Speed**: Due to advanced computing facilities, automation test tools prevail in speed of processing the tests. Automation saves time as software can execute test cases faster than human.

2. **Reduces the tester's involvement in executing tests**: It relieves the testers to do some other work.

3. Repeatability/Consistency: The same tests can be re-run in exactly the same manner eliminating the risk of human errors such as testers forgetting their exact actions, intentionally omitting steps from the test scripts, missing out steps from the test script, all of which can result in either defects not being identified or the reporting of invalid bugs (which can again, be time consuming for both developers and testers to reproduce)

4. Simulated Testing: Automated tools can create many concurrent virtual users/data and effectively test the project in the test environment before releasing the product.

5. Test case design: Automated tools can be used to design test cases also. Through automation, better coverage can be guaranteed than if done manually.

6. Reusable: The automated tests can be reused on different versions of the software, even if the interface changes.

7. **Avoids human mistakes:** Manually executing the test cases may incorporate errors. But this can be avoided in automation testing.

8. Internal Testing: Testing may require testing for memory leakage or checking the coverage of testing. Automation can done this easily.

9. Cost Reduction: If testing time increases cost of the software also increases. Due to testing tools time and therefore cost is reduced.

(e) Draw the diagram of Defect */* Bug life cycle and explain its process.
(Diagram-2Marks, Explanation-2Marks)

Ans: Defect/Bug Life cycle:

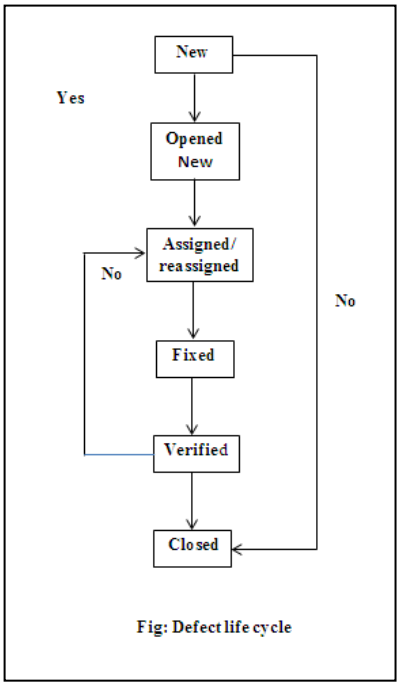
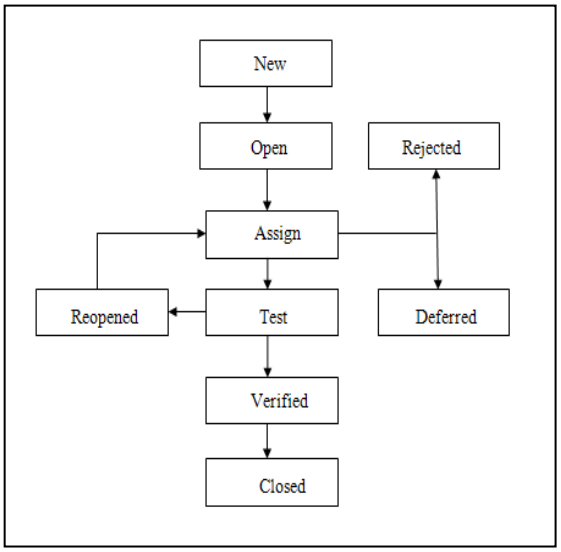


Fig: Defect life cycle

OR



1. **New:** When a defect is logged and posted for the first time. It's state is given as new.
2. **Assigned:** After the tester has posted the bug, the lead of the tester approves that the bug is genuine and he assigns the bug to corresponding developer and the developer team. It's state given as assigned.
3. **Open:** At this state the developer has started analyzing and working on the defect fix.
4. **Fixed:** When developer makes necessary code changes and verifies the changes then he/she can make bug status as 'Fixed' and the bug is passed to testing team.
5. **Pending retest:** After fixing the defect the developer has given that particular

code for retesting to the tester. Here the testing is pending on the testers end. Hence its status is pending retest.

6. **Retest:** At this stage the tester do the retesting of the changed code which developer has
given to him to check whether the defect got fixed or not.
7. **Verified:** The tester tests the bug again after it got fixed by the developer. If the bug is not present in the software, he approves that the bug is fixed and changes the status to "verified".
8. **Reopen:** If the bug still exists even after the bug is fixed by the developer, the tester changes the status to "reopened". The bug goes through the life cycle once again.
9. **Closed:** Once the bug is fixed, it is tested by the tester. If the tester feels that the bug no longer exists in the software, he changes the status of the bug to "closed". This tate means that the bug is fixed, tested and approved.
10. **Duplicate:** If the bug is repeated twice or the two bugs mention the same concept of the bug, then one bug status is changed to "duplicate**".**
11. **Rejected:** If the developer feels that the bug is not genuine, he rejects the bug. Then the state of the bug is changed to "rejected".
12. **Deferred:** The bug, changed to deferred state means the bug is expected to be fixed in next releases. The reasons for changing the bug to this state have many factors. Some of them are priority of the bug may be low, lack of time for the release or the bug may not have major effect on the software.
13. **Not a bug:** The state given as "Not a bug" if there is no change in the functionality of the application. For an example: If customer asks for some change in the look and field of the application like change of color of some text then it is not a bug but just some change in the looks of the application.

(f) Draw classification of White Box testing. Explain any one type of White Box testing in detail.

(Diagram – 1 Mark, Explanation of any one type- 3 Marks)

Ans:

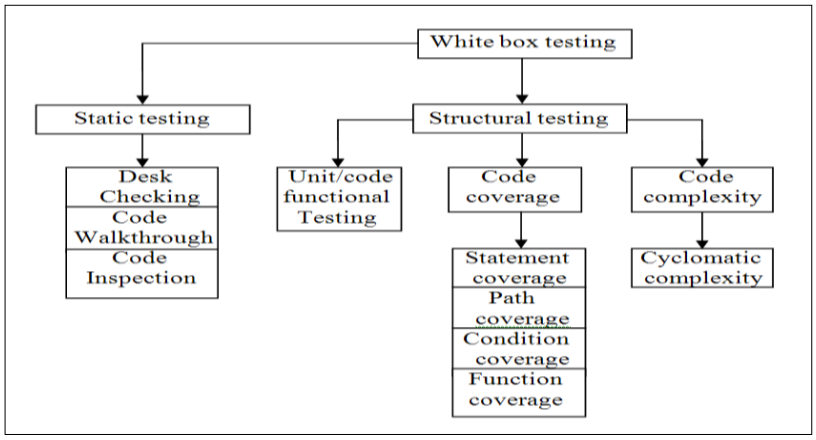


Fig: White Box testing

White Box Testing is classified into:1. Static and 2. Structural testing as shown in the above diagram.

1. **Static Testing**: In static white box testing code has to be reviewed. There rae three methods.
   a. Desk checking: This is usually done by the author of code. The verification of the code is done by comparing the code with the design or specifications to make sure that the code does what it is supposed to do and effectively.

      Disadvantages:
      1. Developer is not the best person to detect his own code.
      2. Developers generally prefer to write new code rather than do any form of testing.
      3. This is person dependent and informal.

   b. Code Walkthrough: Group oriented method. The group of people looks at the program and reviews it. The author has to explain the logic and answers the questions.
   c. Formal Inspection:
      In this:

1. Through preparation before an inspection/review
2. Enlisting multiple diverse views.
3. Assigning specific roles to the multiple participants
4. Going sequentially through the code in a structured manner.
   There are four roles in this.

1. Author of the code.
2. Moderator who is expected to formally run the inspection according to the process.
3. Inspectors are the people who actually provides review comments for the code.
4. Scribe: who takes detail notes during the inspection meeting and circulates them to the inspection team after the meeting.

2. **Structural Testing**: It takes into account the code, code structure, internal design and how they are coded.
   Structural testing is further classified as: a. Unit/code functional testing
   b.Code coverage
   c. Code complexity

a. Unit/code functional testing:

- Initially developer can perform certain obvious tests, knowing the input variables and the corresponding expected output variables.
- For complex modules the developer can build debug version of the product by putting intermediate print statements and making sure the program is passing through the right loops.
- Another approach is to run the product under a debugger or Integrated Development Environment(IDE)

b. Code Coverage Testing: It is made up of the following types of coverage:

1. Statement coverage: It refers to writing test cases that execute each of the program statements.
2. Path Coverage: In this ,the program is split into number of distinct paths.A program can start from the beginning and take any of the paths to its completion. Every path should be tested.
3. Condition Coverage: All condition statements should be tested.
4. Function Coverage: This is to identify how many program functions are covered by test cases.

c. Code Complexity: Cyclomatic complexity is metric that quantifies the complexity of a program and provides answers to the questions such as

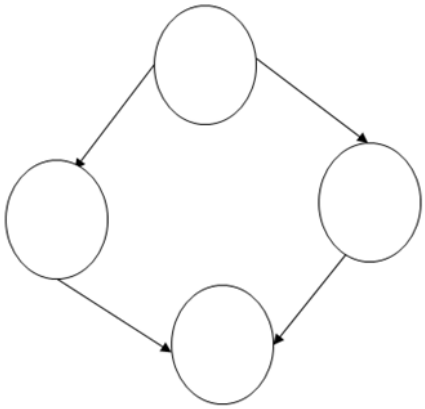
1. Which of the paths are independent? If two paths are not independent then we may be able to minimize the number of tests.
2. Is there an upper bound on the number of tests that must be run to ensure that all the statements have been executed at least once?

   In this a program is represented as flow graph. A flow graph consists of nodes and edges.

   Cyclomatic complexity=Number of predicate nodes(P)+1

   Or

   Cyclomatic complexity=Edges(E)-Nodes(N)+2



In the above flow graph: No. of independent path=2, No. of edges E=4 , No. of nodes N=4

Cyclomatic complexity=E-N+2=4-4+2=2 **or**

Cyclomatic complexity=P+1=1+1=2

3. Attempt any FOUR: **Marks 16**

(a) Describe how to identify responsibilities in testing.
(Explanation-4 Marks)

Ans: A testing project requires different people to play different roles. There are roles of test engineers, test leads and test managers. There is also role definition on the dimensions of the modules being tested or the type of testing. These different roles should complement each other. The different role definition should.

1.Ensure there is clear accountability for a given task, so that each person knows what he or she has to do,
2.Clearly list the responsibilities for various functions to various people, so that everyone knows how his or her work fits into the entire project.
3.Complement each other, ensuring no one steps on an others' toes
4.Supplement each other, so that no task is left unassigned.

Role definition should not only address technical roles, but also list the management and reporting responsibilities. This includes frequency, format and recipients of status reports and other project-tracking mechanism.

(b) Illustrate process of equivalence partitioning with example.
(Definition-1Mark, Example-3 Marks)

Ans: Equivalence partitioning is a software technique that involves identifying a small set of representative input values that produce as much different output condition as possible. This reduces the number of permutation & combination of input, output values used for testing, thereby increasing the coverage and reducing the effort involved in testing.
The set of input values that generate one single expected output is called a partition. When the behavior of the software is the same for a set of values, then the set is termed as equivalence class or partition.

Example: An insurance company that has the following premium rates based on the age group.

A life insurance company has base premium of $0.50 for all ages. Based on the age group, an additional monthly premium has to pay that is as listed in the table below. For example , a person aged 34 has to pay a premium=$0.50 +$ 1.65=$2.15

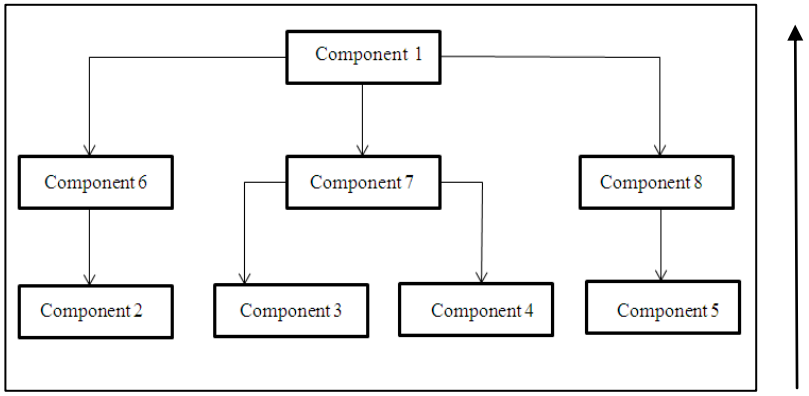
Age group	Additional Premium
Under 35	$ 1.65
35-59	$ 2.87
60+	$ 6.00

Based on the equivalence portioning technique, the equivalence partitions that are based on age are given below:

Below 35 years of age (valid input)
Between 35 and 59 years of age(valid input)
Above 6 years of age (valid input)
Negative age (invalid input)
Age as 0(invalid input)
Age as any three-digit number(valid input)

(c) State process of Bi-directional Integration testing with two advantages and two disadvantages.
(Explanation-2 Marks, For any two advantages- 1 Mark, For any two Disadvantages-1 Mark)

Ans: Bi- directional integration is a combination of the top-down & bottom-up integration approaches used together to derive integration steps.



As shown in fig , assume that the software components become available in the order mentioned by the component numbers. The individual components 1,2 ,3,4 and 5 are tested separately and bi-directional integration performed initially with the use of stubs and drivers. Drivers are used to provide upstream connectivity while stubs provide downstream connectivity .A driver is a function which redirects the requests to some other components and stubs simulate the behavior of missing components. After the functionality of these integrated components are tested, the drivers and stubs are discarded. Once components 6,7 and 8 become available, the integration methodology then focus only on those components , as there are the components which need focus and are new. This approach is also called ***"Sandwich Integration".***

Advantages:

1. Sandwich approach is useful for very large projects having several subprojects.
2. Both Top-down and Bottom-up approach starts at a time as per development schedule. Units are tested and brought together to make a system .Integration is done downwards.

Disadvantages:

1. It require very high cost for testing because one part has Top-down approach while another part has bottom-up approach.
2. It cannot be used for smaller system with huge interdependence between different modules. It makes sense when the individual subsystem is as good as complete system.

(d) Describe how to perform security and performance testing.
(Explanation of security testing-2 Marks, Explanation of performance testing-2 Marks)

[Note: Also refer answer of Q.6(e)]

Ans: Security Testing:

Testers must use a risk-based approach, By identifying risks and potential loss associated with those risks in the system and creating tests driven by those risks, the testers can properly focus on areas of code in which an attack is likely to succeed. Therefore risk analysis at the design level can help to identify potential security problems and their impacts. Once identified ranked, software risks can help guide software security.

Performance testing: Performance testing is intended to find whether the system meets its performance requirements under normal load or abnormal level of activities. Normal load must be defined by the requirement statement defined by the customer and system design implements them. Performance criteria must be expressed in numerical terms. Design verification can help in determining whether required measures have been taken to meet performance requirements or not. This is one area where verification does not work to that much extents and one needs to test it by actually performing the operation on the system.

(e) Define metrics and measurements. Explain need of software measurement.
(Definition-2 Marks, Any two Need-2 Marks)

Ans: Metrics & measurement: Metrics is a relative measurement of status of process or product in terms of two or more entities taken together for comparison.
Measurements are key element for controlling software engineering processes.

Need of software measurements:
1. **Understanding**: Metrics can help in making the aspects of process more visible, thereby giving a better understanding of the relationship among the activities and entities they affect.
2. **Control**: Using baselines, goals and an understanding of the relationships, we can predict what is likely to happen and correspondingly, make appropriate changes in the process to help meet the goals.
3. Improvement: By taking corrective actions and making appropriate changes, we can improve a product. Similarly, based on the analysis of a project, a process can also be improved.

4. [A] Attempt any THREE: **Marks 12**

(a) How to prepare a test plan? Explain in detail.
(Any four explanation-4 Marks)

Ans: The test plan acts as the anchor for the execution, tracking and reporting of the entire testing project and covers.
1.Preparing test plan:
1. What needs to be tested – the scope of testing, including clear identification of what will be the tested & what will not be tested.
2. How the testing is going to be performed – breaking down the testing into small and manageable tasks and identifying the strategies to be used for carrying out the tasks.
3. What resources are needed for testing- computer as well as human resources.
4. The time lines by which the testing activities will be performed.
5. Risks that may be faced in all of the above, with appropriate mitigation and contingency plans.

2.Scope management: It entails:
1. Understanding what constitutes a release of product.
2. Breaking down the release into features.
3. Prioritizing the feature of testing.
4. Deciding which features will be tested & which will not be

3.Deciding Test approach/ strategy: This includes identifying.

1. What type of testing would use for testing functionality?
2. What are the configurations for testing features?
3. What integration testing would you do to ensure these features work together?
4. What "non-functional" tests would you need to do?

4.Setting up criteria for testing: Some of the typical suspension criteria include:

1. Encountering more than a certain numbers of defects, causing frequent stoppage of testing activity.
2. Hitting show stoppers that prevent further progress of testing.

5.Identifying responsibilities, staffing & Training needs: The next aspect of planning is who part of it. Identifying responsibilities , staffing & training needs addresses this aspect.
6.Identifying Resource Requirement: As a part of planning for a testing project, the project manager should provide estimate for the various h/w & s/w resources required.
7.Identifying Test Deliverables: It includes : test plan itself, test case design specification, test cases , test logs & test summary report
8.Testing task: Size and Effort estimation: This gives estimation in terms of size, effort & schedule of testing project.

(b) Explain static and dynamic testing tools in details.
(Static tools-2 Marks, Dynamic tools- 2 Marks)

Ans: Static testing tools are used during static analysis of a system. Static testing tools are used throughout a software development life cycle, e.g , tools used for verification purposes. There are many varieties of static testing tools used by different people as per the type of system being developed.

Code complexity measurement tools can be used to measure the complexity of a given code. Similarly, data-profiling tools can be used to optimize a database. Code-profiling tools can be used to optimize code. Test-generators are used for generating a test plan form code. Syntax-checking tools are used to verify correctness of code.

Dynamic testing tools are used at different levels of testing starting from unit testing & which may go up to system testing & performance testing. These tools are generally used by tester. There are many different tools used for dynamic testing. Some of the areas covered by testing tools are:

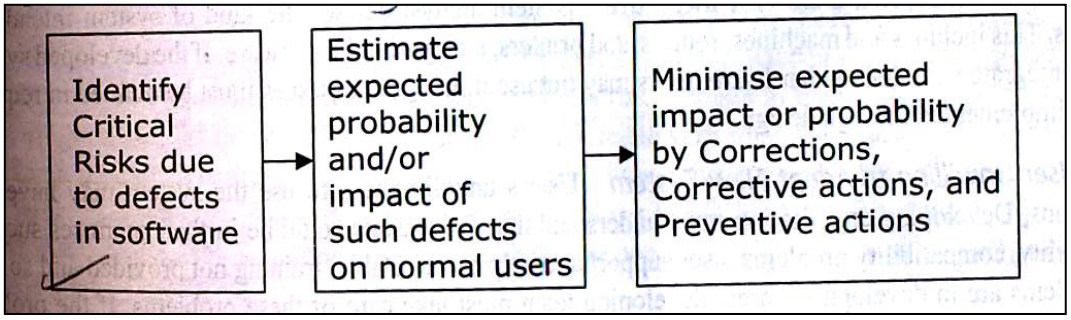
1. Regression testing using automated tools.
2. Defect tracking and communication systems used by tracking & communication.

Performance, Load, stress-testing tools.

(c) Illustrate defect prevention process of Defect fixing process with diagram.
(Diagram-2 Marks, Explanation-2 Marks)

[Note: Any other relevant answer shall be considered]

Ans: Defect prevention is a cycle of risk analysis and actions based on its ranking. Defects found in verification & validation must be realistic and represent the possible areas of improvement in development and testing processes. Defect must be governed by probability of happening & severity of the defect in terms of impact on the users and detection ability of occurrence of such defect prevention mechanism. The process of risk analysis & defect prevention is shown in fig.



(d) Explain test deliverables in details.
(Explanation-4 Marks)

Ans: The test plan also identifies the deliverables that should come out of the test cycle/testing activity. The Deliverables includes the following, all reviewed and approved by the appropriate people.

1. The test plan itself(master test plan and previous other test plans for the project)
2. Test case design specifications.
3. Test cases, including any automation that is specified in the plan.
4. Test logs produced by running the tests.
5. Test summary reports

[B] Attempt any ONE: **Marks 6**

(a) Explain V-model with labelled diagram.

(Diagram-2 Marks, Explanation of Verification Phase- 2 Marks, Explanation of Validation Phase-2 Marks)

Ans: V model means verification and validation model. It is sequential path of execution of processes. Each phase must be completed before the next phase begins.

Under V-model, the corresponding testing phase of the development phase is planned in parallel. So there is verification on one side of V & validation phase on the other side of V.

Verification Phase:

1. **Overall Business Requirement:** In this first phase of the development cycle , the product requirements are understood from customer perspective. This phase involves detailed communication with the customer to understand his expectations and exact requirements. The acceptance test design planning is done at this stage as business requirements can be used as an input for acceptance testing.
2. **Software Requirement**: Once the product requirements are clearly known, the system can be designed. The system design comprises of understanding & detailing the complete hardware , software & communication set up for the product under development. System test plan is designed based on system design. Doing this at earlier stage leaves more time for actual test execution later.
3. **High level design:** High level specification are understood & designed in this phase. Usually more than one technical approach is proposed & based on the technical & financial feasibility, the final decision is taken. System design is broken down further into modules taking up different functionality.
4. **Low level design:** In this phase the detailed integral design for all the system modules is specified. It is important that the design is compatible with the other modules in the system & other external system. Components tests can be designed at this stage based on the internal module design,
5. **Coding:** The actual coding of the system modules designed in the design phase is taken up in the coding phase. The base suitable programming language is decided base on requirements. Coding is done based on the coding guidelines & standards.

Validation:

1. **Unit Testing:** Unit testing designed in coding are executed on the code during this validation phase. This helps to eliminate bugs at an early stage.
2. **Components testing:** This is associated with module design helps to eliminate defects in individual modules.
3. **Integration Testing:** It is associated with high level design phase & it is performed to test the coexistence & communication of the internal modules within the system

4. **System Testing:** It is associated with system design phase. It checks the entire system functionality & the communication of the system under development with external systems. Most of the software & hardware compatibility issues can be uncovered using system test execution.

Acceptance Testing: It is associated with overall & involves testing the product in user environment. These tests uncover the compatibility issues with the other systems available in the user environment. It also uncovers the non-functional issues such as load & performance defects in the actual user environment.

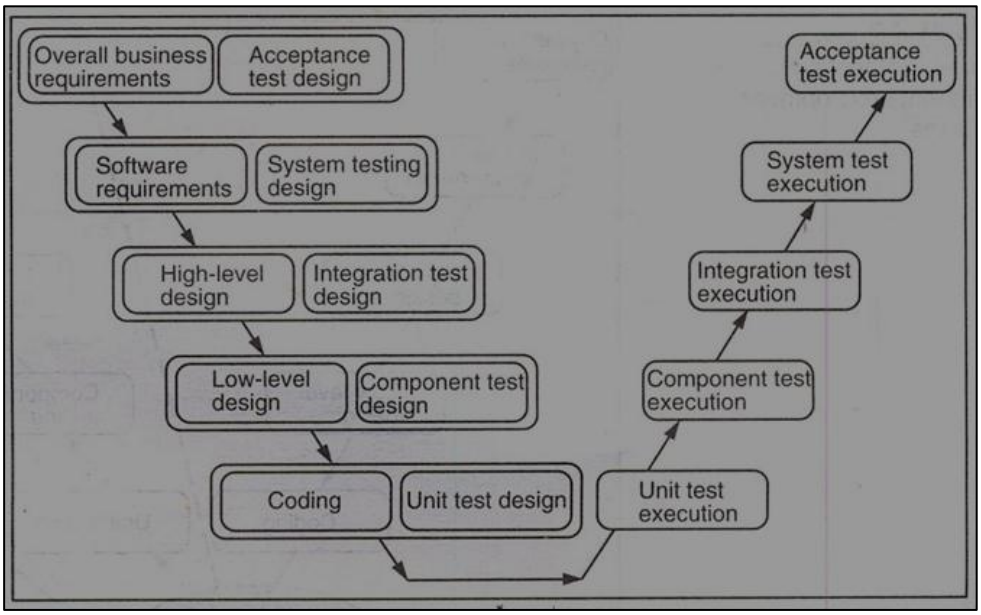
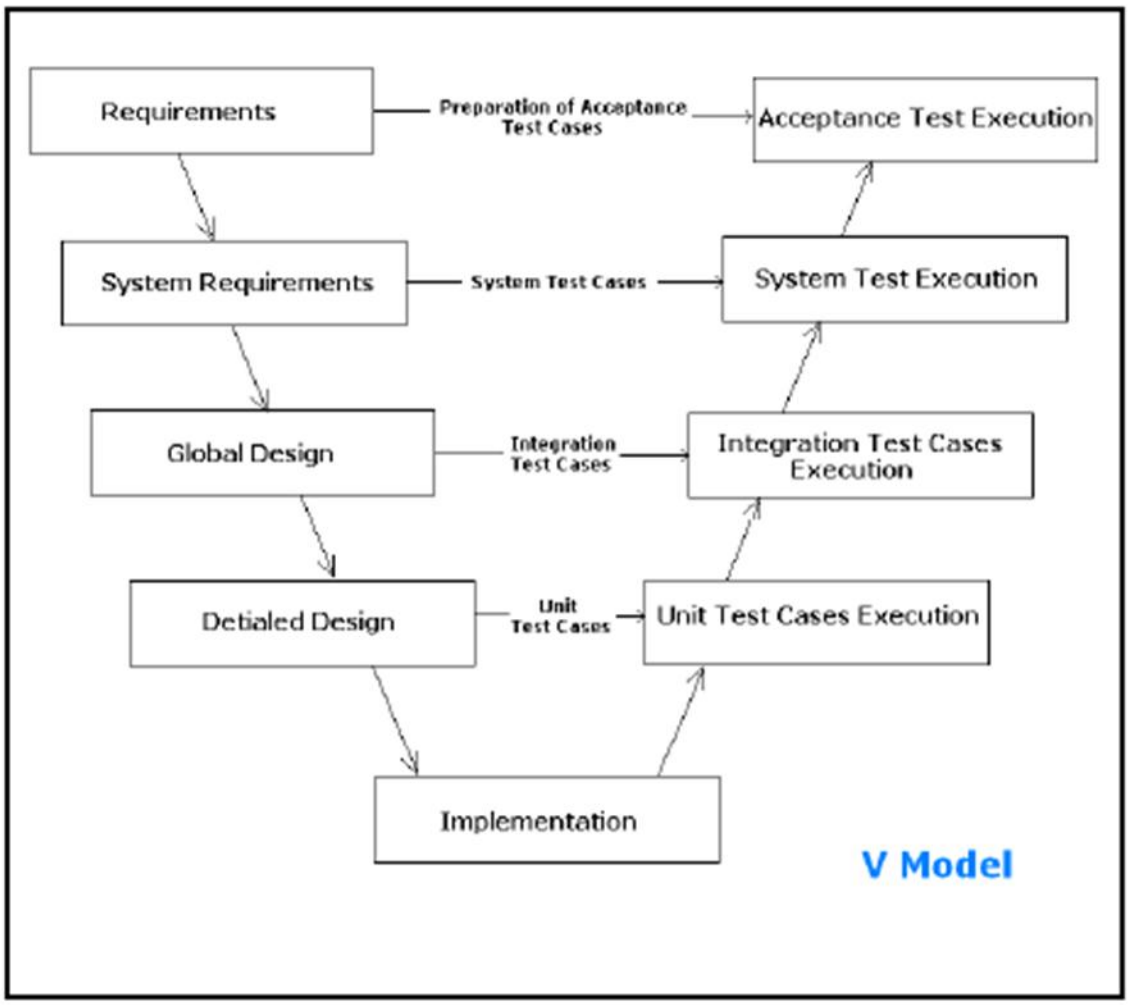


Fig: V-Model

OR


Requirements
Preparation of Acceptance Test Cases
Acceptance Test Execution
System Requirements
System Test Cases
System Test Execution
Global Design
Integration Test Cases
Integration Test Cases Execution
Detialed Design
Unit Test Cases
Unit Test Cases Execution
Implementation
V Model

(b) State two objectives of user documentation testing? Mention any two benefits of user documentation testing.
(Any two objectives -3 Marks, Any two benefits-3 Marks)

Ans: Objectives:
1.To check if what is stated in the document is available in the product.
2.To check if what is there in the product is explained correctly in the document.

Benefits:
1.User documentation testing aids in highlighting problems over looked during reviews.
2.High quality user documentation ensures consistency of documentation & product, thus minimizing possible defects reported by customer. It also reduces the time taken for each support call.
3.Result in less difficult support call. When customer faithfully follows the instruction given in a document but is unable to achieve to desire result, it is frustrating and often this frustration shows up on the support staff. Ensuring that a product is tested to work as per the document and that it works correctly contributes to better customer satisfaction and better morale of support staff.
4. New programmers and testers who join a project group can use the documentation to learn the external functionality of the product.
Customers need less training and can proceed more quickly to advanced training & product usage if the user documentation is of high quality & is consistent with the product.
Thus high-quality user documentation can result in a reduction of overall training costs for user organization.

5. Attempt any TWO: Marks 16

(a) Describe test reporting in detail and how to prepare a summary report.
(Test reporting explanation-4Marks, Test summary report -4Marks)

Ans:
Test reporting is a means of achieving communication through the testing cycle. There are 3 types of test reporting.

1. **Test incident report:**
A test incident report is communication that happens through the testing cycle as and when defects are encountered .A test incident report is an entry made in the defect repository each defect has a unique id to identify incident .The high impact test incident are highlighted in the test summary report.

2. Test cycle report:

A test cycle entails planning and running certain test in cycle , each cycle using a different build of the product .As the product progresses through the various cycles it is expected to stabilize.

Test cycle report gives

1. A summary of the activities carried out during that cycle.
2. Defects that are uncovered during that cycle based on severity and impact
3. Progress from the previous cycle to the current cycle in terms of defect fixed
4. Outstanding defects that not yet to be fixed in cycle
5. Any variation observed in effort or schedule

3 Test summary report:

The final step in a test cycle is to recommend the suitability of a product for release. A report that summarizes the result of a test cycle is the test summary report.

There are two types of test summary report:

1.Phase wise test summary ,which is produced at the end of every phase

2. Final test summary report .

A Summary report should present

1. Test Summary report Identifier
2 Description
   Identify the test items being reported in this report with test id
3 Variances
   Mention any deviation from test plans, test procedures, if any.
4 Summary of results
   All the results are mentioned here with the resolved incidents and their solutions.
5 Comprehensive assessment and recommendation for release should include
   Fit for release assessment and recommendation of release

(b) With the help of example explain Boundary Value Analysis.
(Explantion-4 Marks, Example-4 Marks)

Ans: Most of the defects in software products hover around conditions and boundaries. By conditions, we mean situations wherein, based on the values of various variables, certain actions would have to be taken. By boundaries, we mean "limits" of values of the various variables.

- This is one of the software testing technique in which the test cases are designed to include values at the boundary. If the input data is used within the boundary value

limits, then it is said to be Positive Testing. If the input data is picked outside the boundary value limits, then it is said to be Negative Testing.

- 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.
- 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.
- Same examples of Boundary value analysis concept are:
  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.
- **For Example:** A system can accept the numbers from 1 to 10 numeric values. All othernumbers are invalid values. Under this technique, boundary values 0,1,2, 9,10,11 can be tested.
- Another Example is in exam has a pass boundary at 40 percent, merit at 75 percent and
  distinction at 85 percent. The Valid Boundary values for this scenario will be as follows:
  49, 50 - for pass
  74, 75 - for merit
  84, 85 - for distinction
  Boundary values are validated against both the valid boundaries and invalid boundaries. The Invalid Boundary Cases for the above example can be given as follows
  0 - for lower limit boundary value
  101 - for upper limit boundary value
- Boundary value analysis is a black box testing and is also applies to white box testing. Internal data structures like arrays, stacks and queues need to be checked for boundary or limit conditions; when there are linked lists used as internal structures, the behavior of the list at the beginning and end have to be tested thoroughly.
- Boundary value analysis help identify the test cases that are most likely to uncover defects.

(c) Explain following concept related to web-based testing:

(1) Usability testing

(2) Compatibility testing

(Usability testing-4Marks, Compatibility testing-4Marks)

Ans:

Usability testing related to web-based testing:

The presentation design emphasizing the interface between user and web application gives rise to usability testing.

The general guideline for usability testing

1. Do not force users to remember key information across document
2. Present information in a natural and logical order
3. All possible options like menus, links or buttons on web pages should be visible and accessible from all web pages.
4. Check that link are active such that there are no erroneous or misleading links
5. Indicate similar concept through identical terminal and graphics.
6. Provide understandable instruction where useful.
7. The user should not get irritated while navigating through the web application .eliminate information which is irrelevant or distracting.
8. Content writer should not mix the topics of information. There should be clarity in the information being displayed
9. Organize information hierarchically, with more general information appearing before more specific details encourage the user to delve as deeply as needed, but to stop whenever sufficient information has been received

Compatibility testing related to web-based testing:

Web application may be put in different environment when the user is using them in production. Server may be in different hardware, software or operating system environment than the recommended one. Client browsers may differ significantly from the expected environmental variable. Testing must ensure that performance is maintained on the range of hardware and software configuration and user must be adequately protected in case of configuration mismatch.

The general guideline for compatibility testing:

1. There are number of different browsers and browser options. The web application has to be designed to be compatible for majority of the browser.
2. The graphics have and other objects on website have to be tested on multiple browser.

3. There are different versions of HTML. Also there are other code has to be tested like Java,Javascript,ActiveX etc.
4. Test your web application on different operating systems like windows,Unix,MAC,Linux.

6. Attempt any FOUR of following: Marks 16

(a) What is defect management? Give defect classification in detail.
(Definition -1Mark, Explantion-3Marks)

Ans: It is the process of It enhances quality by adding value to the most important attributes of software like reliability, maintainability, efficiency and portability.

Defect Classification:

Requirements and specification defect: Requirement related defects arise in a product when one fails to understand what is required by the customer. These defects may be due to customer gap, where the customer is unable to define his requirements, or producer gap, where developing team is not able to make a product as per requirements. Defects injected in early phases can persist and be very difficult to remove in later phases. Since any requirements documents are written using natural language representation, there are very often occurrences of ambiguous, contradictory, unclear, redundant and imprecise requirements. Specifications are also developed using natural language representations.

Design Defects: Design defects occur when system components, interactions between system components, interactions between the outside software/hardware, or users are incorrectly designed. This covers in the design of algorithms, control, logic/ data elements, module interface descriptions and external software/hardware/user interface descriptions. Design defects generally refer to the way of design creation or its usage while creating a product. The customer may or may not be in a position to understand these defects, if structures are not correct. They may be due to problems with design creation and implementation during software development life cycle.

Coding Defects: Coding defects may arise when designs are implemented wrongly. If there is absence of development/coding standards or if they are wrong, it may lead to coding defects.Coding defects are derived from errors in implementing the code. Coding defect classes are closely related to design defect classes especially if pseudo code has been used for detailed design. Some coding defects come from a failure to understand programming language constructs, and miscommunication with the designers. Others may have transcription or omission origins. At times it may be difficulty to classify a defect as a design or as a coding detect.

Testing Defect: Testing defect are defects introduced in an application due to wrong testing , or defects in the test artifact leading to wrong testing. Defects which cannot be

reproduced , or are not supported by requirement or are duplicate may represent a false call .In this defects includes

1. Test-design defect : test-design defect refers to defects in test artifacts. there can be defects in test plans, test scenarios, test cases and test data definition which can lead to defect in software.
2. Test-environment defect: this defect may arise when test environment is not set properly. test environment may be comprised of hardware, software, simulator and people doing testing.
3. Test-tool defects: any defects introduced by a test tool may be very difficult to find and resolve, as one may have to find the defect using manual test as against automated tools.

(b) Give difference between Quality Assurance and Quality Control. (Any Four)
(Any Four differences – 4Marks)

Ans:

Quality Assurance	Quality Control
Process oriented activities.	Product oriented activities.
QA is the process of managing for quality.	QC is used to verify the quality of the output
They measure the process, identify the deficiencies/weakness and suggest improvements.	They measure the product, identify the deficiencies/weakness and suggest improvements.
Relates to all products that will ever be created by a process	Relates to specific product
Activities of QA are Process Definition and Implementation, Audits and Training	Activities of QC are Reviews and Testing
Verification is an example of QA	Validation/Software Testing is an example of QC
Preventive activities.	It is a corrective process.
Quality assurance is a proactive process	Quality control is a reactive process.
QA is a managerial tool	QC is a corrective tool

(c) State how to minimize risk impact while estimating defect.
(For each point -1 Mark)

Ans: Risk is a product of probability, impact and detection ability.
Risk minimization has three different methods of handling its probability, impact or detection ability. The decisions may be driven by organization policy, values, cost-benefit analysis etc. Minimization of problem due to risk happens in the following manner

- **Eliminate Risk**: Elimination of risk involves taking steps to remove risk from the root. Risk's probability is reduced to almost 'O' by removing the causes of risk, so that the risk must not happen at all, or the organization user may be protected from the possible losses arising due to risk. Preventive controls can eliminate the probability of risk to a large extent. Preventive controls are management-decided controls. Preventive controls are applied, if the probability of occurrence is very high. Preventive controls are useless, if the probability of happening is already negligible.
- **Mitigation of Risk**: Actions initiated by an organization to minimize the possible damage due to realization of risk are considered 'mitigation actions'. Mitigation actions are planned by an organization/project so that if the risk is realized, the impact due to it can be reduced to minimum possible. The corrective controls used from the mitigation action. Corrective controls may be auto-corrective or suggestive.
- **Detection Ability Improvements**: Impact of a risk is more, if it catches the user unprepared. If people are aware of the risks, they can be well prepared to handle them. Generally, detective controls are used to increase the visibility towards risks. Sometimes, detective controls give threshold to corrective controls.
- **Contingency Planning**: Contingency planning refers to the actions initiated by an Organization, when preventive or corrective actions fail and risk actually occurs. They are Previously planned ways of tackling risks when all other planned activities for reducingProbability and impact of the risk fail, and the risk becomes reality.

(d) Write 4 test cases for user login form.

(Four test cases – 4 Marks)

Ans:

TC_Id	TC_name	Steps	Input data	Expected result	Actual Result	Status
TC_01	User name	Enter the username in alphanumeric alphabets A-Z Number 0-9	"abc123"	It should accept the username	It is accepted username	Pass
TC_02	Password	Enter the password in alphanumeric alphabets A-Z Number 0-9	"abc123"	It should accept the password	It is accepted password	pass
TC_03	Submit	1.After valid username and password 2. Click on submit button		It should goes to next page	It is going to next page	pass
TC_04	Cancel	Click on cancel button		It should remain in login page with blank fields	It shows login page with blank fields	pass

(e) Describe following testing with example.
(1) Security testing
(2) Performance testing
(For each testing – 2 Marks)
***[Note: Also r*efer answer of Q.3 (d)]**

Ans: Security testing

- It is a type of non-functional testing.
- Security testing is basically a type of software testing that's done to check whether the application or the product is secured or not. It checks to see if the application is vulnerable to attacks, if anyone hack the system or login to the application without any authorization.
- It is a process to determine that an information system protects data and maintains functionality as intended.
- The security testing is performed to check whether there is any information leakage in the sense by encrypting the application or using wide range of software's and hardware's and firewall etc.
- Software security is about making software behave in the presence of a malicious attack.
- The six basic security concepts that need to be covered by security testing are: confidentiality, integrity, authentication, availability, authorization and non-repudiation.

2) **Performance Testing**

- Performance testing is testing that is performed, to determine how fast some aspect of a system performs under a particular workload.
- It can serve different purposes like it can demonstrate that the system meets performance criteria.
- **Stress Testing:**
- It is a type of non-functional testing.
- It involves testing beyond normal operational capacity, often to a breaking point, in order to observe the results.
- It is a form of software testing that is used to determine the stability of a given system.
- It put greater emphasis on robustness, availability, and error handling under a heavy load, rather than on what would be considered correct behavior under normal circumstances.
- The goals of such tests may be to ensure the software does not crash in conditions of insufficient computational resources (such as memory or disk space).

- **Load Testing:**
- Load testing is a type of non-functional testing.

- A load test is type of software testing which is conducted to understand the behavior of the application under a specific expected load.
- Load testing is performed to determine a system's behavior under both normal and at peak conditions.
- It helps to identify the maximum operating capacity of an application as well as any bottlenecks and determine which element is causing degradation. E.g. If the number of users are increased then how much CPU, memory will be consumed, what is the network and bandwidth response time.
- Load testing can be done under controlled lab conditions to compare the capabilities of different systems or to accurately measure the capabilities of a single system.

- **Examples of load testing include**:
- Downloading a series of large files from the internet.
- Running multiple applications on a computer or server simultaneously.