MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2013 Certified)

_____

WINTER – 2023 EXAMINATION
Model Answer – Only for the Use of RAC Assessors

Subject Name: Software Testing   **Subject Code:** 22518

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model 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 judgement 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.
8) As per the policy decision of Maharashtra State Government, teaching in English/Marathi and Bilingual (English + Marathi) medium is introduced at first year of AICTE diploma Programme from academic year 2021-2022. Hence if the students in first year (first and second semesters) write answers in Marathi or bilingual language (English +Marathi), the Examiner shall consider the same and assess the answer based on matching of concepts with model answer.

Q. No.	Sub Q. N.	Answer	Marking Scheme
1		**Attempt any <u>FIVE</u> of the following:**	**10 M**
	a)	**Compare Verification and validation. (any 2 points)**	**2 M**
	Ans	<table><tr><th>Verification</th><th>Validation</th></tr><tr><td>Are we building the system, right?</td><td>Are we building the right system?</td></tr><tr><td>Verification is the process of evaluating products of a development phase to find out whether they meet the specified requirements.</td><td>Validation is the process of evaluating software at the end of the development process to determine whether software meets the customer expectations and requirements.</td></tr></table>	comparison of verification and validation: 2 points: 2 M, 1 M each

		Execution of code is not comes under **Verification**.	Execution of code is comes under **Validation**.	
		Verification is carried out before the Validation.	**Validation** activity is carried out just after the Verification.	
		Cost of errors caught in **Verification** is less than errors found in Validation.	Cost of errors caught in **Validation** is more than errors found in Verification.	
		It is basically manually checking the of documents and files like requirement specifications etc.	It is basically checking of developed program based on the requirement specifications documents & files.	
	b)	**Define failure, error, fault, bug.**		**2 M**
	Ans	**Failure:** It is the inability of a system or component to perform the required function according to its specification. OR External behavior is incorrect. **Error:** Refers to difference between Actual Output and Expected output. OR An error is a human action that produces the incorrect result. **Fault:** It is a condition that causes the software to fail to perform its required function. OR Discrepancy in code that causes a failure. **Bug:** The presence of error at the time of execution of the software. OR A bug can be defined as the initiation of error or a problem due to which fault, failure, incident or an anomaly occurs.		Definition of failure, error, fault, bug: ½ M each, total 2 M
	c)	**List the objectives of software testing (any four).**		**2 M**
	Ans	**Objectives of software testing:** 1. **Finding error:** Finding defects which may be created by the programmer while developing the software. 2. **Quality improvement:** Gaining confidence in and providing information about the level of quality. 3. **Creating good test cases**: Good test case is one that has a high probability of finding undiscovered error. 4. **Meets users' requirements:** To make sure that the result meets the business and user requirements. 5. **Satisfying Requirements:** To ensure that it satisfies the BRS that is Business Requirement Specification, and SRS that is System Requirement Specifications. 6. To gain the confidence of the customers by providing them a quality product		2 objectives of software testing: 2 M; 1 M each

d)		**Define driver and stub.**	**2 M**
	Ans	**Driver:** Drivers are dummy modules that are always used to simulate the high-level modules. Drivers are only used when main programs are under construction. Drivers are used in bottom-up integration. **Stub:** Stubs are dummy modules that always used to simulate the low-level modules. Stubs are used when sub programs are under construction. Stubs are used in top-down approach.	Definition of driver: 1 M; definition of stub: 1 M
e)		**What is GUI testing? Give one example.**	**2 M**
	Ans	**GUI Testing:** GUI stands for Graphical User Interface where you interact with the computer using.	GUI Testing: 1 M; any valid

		images rather than text. ● GUI testing is the process of testing the system's Graphical User Interface of the Application Under Test. GUI testing involves checking the screens with the controls like menus, buttons, icons, and all types of bars - toolbar, menu bar, dialog boxes and windows, etc. GUI is what the user sees. A user does not see the source code. The interface is visible to the user. Especially the focus is on the design structure, images that they are working properly or not. **Examples of GUI testing includes:** 1. Check Screen Validations 2. Verify All Navigations 3. Check usability Conditions 4. Verify Data Integrity 5. Verify the object states 6. Verify the date Field and Numeric Field Formats	example: 1 M; total 2 M
f)		**Write any two root causes of defect.**	**2 M**

	Ans	Root causes of defect are: i. Miscommunication of requirements introduces error in code. ii. Lack of design Experience. iii. Lack of coding practice. iv. Unrealistic time schedule for development. v. Multiple changes in the requirements. **OR** **Any other valid answer shall be given marks.**	Two root causes of defect: 2 M; 1 M each
	g)	**Enlist any four software testing tools.**	**2 M**
	Ans	4 software testing tools are: 1. Selenium 2. Test complete 3. LoadRunner 4. Cucumber 5. Quick test professional (QTP) 6. Cypress **(Any valid software testing tool shall be given marks)**	any 4 software testing tools: 2 M; ½ M each
2.		**Attempt any THREE of the following:**	**12 M**
	a)	**State the entry and exit criteria for software testing.**	**4 M**

Ans	**Entry criteria** Entry criteria are the condition or the set of conditions, which should exist or be met to start a process. Some of the conditions or situations, which may be seen as an entry criterion for the initiation of testing activities. • Requirements should be clearly defined and approved. • Test Design and documentation plan is ready. • Availability of the test environment supporting necessary hardware, software, network configuration, settings, and tools for the purpose of test execution. • Testers are trained, and necessary resources are available. • Availability of proper and adequate test data (like test cases). • It depends upon which software development model is used. **Exit criteria.** Exit Criteria is often viewed as a single document concluding the end of a life cycle phase. Some of the conditions or situations which may be seen as an exit criterion for testing activities. • Testing Deadline • Completion of test case execution. • Completion of Functional and code coverage to a certain point. • Bug rates fall below a certain level and no high priority bugs are identified. • Management decision.		entry criteria: 2 M exit criteria: 2 M; total 4 M
b)	**Illustrate process of bi-directional integration testing. State it's two advantages and disadvantages.**		**4 M**

_____

| | Ans | • Bidirectional integration testing strategy is a combination of Top Down and Bottom-up approaches.
• Here, top modules are tested with lower modules at the same time lower modules are integrated with top modules and tested.
• This strategy makes use of stubs as well as drivers.
• bidirectional integration testing is a culmination of both incremental as well as non-incremental integration testing, wherein Bottom-Up approach is focused on middle to top layer, Top-Down approach is concerned about layers from middle to downwards and the Big Bang approach is followed for the middle layer.
• This type of testing combines the advantages of all the three approaches and is mainly used to test large projects.

Advantages:
1. Bidirectional integration testing is very useful for large enterprises and huge projects that further have several subprojects.
2. When development follows a spiral model and the module itself is as large as a system, then one can use Bidirectional integration testing.
3. Top-Down and Bottom-Up approach both start as per development schedule.
4. Units are tested and brought together to make a system.
5. Integration is done downwards.
6. The resources that are required are immense and big teams perform both topdown and bottom-up method of testing at a time or one after the other.
Disadvantages: | Process of Bidirectional integration testing: 2 M; 2 advantages:
1   M;
disadvantages:
1 M;   total 4 M |

| | | 1. As both Top-Down and Bottom-Up approaches are executed on the software, the cost of testing is very high.
2. It cannot be used for smaller systems with huge interdependence between the modules.
3. It only makes sense when the individual subsystem is as good as the completed system.
4. Different skill sets are required for testers at different levels. | |
| | **c)** | **Enlist four attributes of defect. Describe them with suitable example.** | **4 M** |

	Ans	**Attributes of defect:**	4 attributes of defect with example: 4 M; 1 M each
		1) Defect ID: Identifies defect as there are many defects might identified in system. a. i.e. D1, D2, etc.	
		2) Defect Name: Name of defect which explains the defect in brief. a. It must be short but descriptive. i.e. Login error.	
		3) Project Name: Indicates project name in which defect is found e.g.: Library management system	
		4) Module /Sub-module name: for which the defect is found. E.g. Login form	
		5) Phase introduced: Phase of life cycle to which the defect belongs to. e.g. 2	
		6) Phase found: Phase of project when the defect is found is added here. It is used to find defect leakage or stage.	
		7) Defect type: Defines defect type. e.g. security defects, functional defect, GUI defect etc.	
		8) Severity: Declared in test plan, e.g. high medium or low.	
		9) Priority: defines based on how the project decides a schedule to take the defects for fixing. e.g. High, low, Moderate	
		10) Summary: Describes short about the defect.	
		11) Description: Describes it in detail.	
		12) Status: dynamic field, open, assigned, resolved, closed, hold, deferred, or reopened, etc.	
		13) Reported by/ Reported on: Who found defect, and on what date.	
		14) Assigned to: The tester is being assigned to some testing team member.	
	d)	**Describe any four factors for selecting a testing tool.**	**4 M**
	Ans	**4 factors for selection of testing tools are:** **The following factors are important during tool selection:** i. **Assessment of the organization's maturity** (e.g. readiness for change); ii. Identification of the areas within the organization where **tool support will help to improve testing processes;** iii. **Evaluation of tools against clear requirements and objective**	4 factors for selection of testing tools: 4 M; 1 M each

criteria;
iv. **Proof-of-concept to see whether the product works as desired** and meets the requirements and objectives defined for it;
v. **Evaluation of the vendor (training, support and other commercial aspects**) or open-source network of support; vi. **Identifying and planning internal implementation** (including coaching and mentoring for those new to the use of the tool).

OR (SOME CONTENTS FROM FOLLOWING FACTORS ALSO SHALL BE GIVEN MARKS)

The industry experts have suggested following four major criteria's for selection of testing tools.
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 all the requirements of a given product or a given organization.

• Evaluating different tools for different requirements involve significant effort, money, and time. • 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.

1. 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.

• Different phases of a life cycle have different quality-factor requirements. Tools required at each stage may differ significantly.

2. 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.

_____

• If the testers do not have proper training and skill, then they may not be able to work effectively.

3. 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. • Select affordable tools. Cost and benefits of various tools must be compared before making final decision.	
3.		**Attempt any THREE of the following:**	**12 M**
	a)	**Differentiate between Alpha Testing and Beta Testing. (any four points)**	**4 M**
	Ans	**Differences Between Alpha Testing and Beta Testing:**	1 M each Differentiations

Alpha Testing	Beta Testing
Alpha Testing performed at developer's site.	Beta testing is performed at a client location or end user of the product.
Alpha testing involves both the white box and black box techniques.	Beta Testing typically uses Black Box Testing.
Alpha testing performed by Testers who are usually internal employees of the organization.	Beta testing is performed by Clients or End Users who are not employees of the organization.
Alpha testing requires a lab environment or testing environment	Beta testing doesn't require any lab environment or testing environment. The software is made available to the public and is said to be real time environment
Alpha testing is to ensure the quality of the product before moving to Beta testing	Beta testing also concentrates on the quality of the product, but gathers users input on the product and ensures that the product is ready for real time users.
Critical issues or fixes can be addressed by developers immediately in Alpha testing.	Most of the issues or feedback is collected from Beta testing will be implemented in future versions of the product.
Long execution cycle may be required for Alpha testing.	Only a few weeks of execution are required for Beta testing

	b)	**Prepare Test Plan for Notepad Application. (Windows based)**	**4 M**

Ans					Consider any 8 points
	1				
		Test Plan Identifier		TP_10	
	2	**Introduction:**		The purpose of this document is to create and test plan for edit functionality of notepad. The purpose of testing this program is to check the correct operation of its functionality	

				and ease of use.	
		3	**Test Items**	Working with notepad	
		4	**Features to be tested.**	• Undo • Cut /Copy • Paste • DELETE • Find • Goto	
		5	**Approach**	• On the test object: o functional o non-functional • According to the requirements o positive o negative • By degree of preparedness - intuitive testing (ad hoc)	
		6	**Item Pass/Fail Criteria:**	All test cases with high priority are closed with the result - pass. The test coverage is checked and sufficient, where the criterion of sufficiency is not less than 99% of the coverage of requirements by tests. The test report was compiled and approved by the team lead and customer.	

7	**Suspension Criteria:**	The appearance and entering the bug-tracking system of blocking bugs.
	Resumption Criteria:	Closing the blocking bug in the bug tracking system
8	**Test Deliverables**	Test plan, test case specification, test case, test summary report
9	**Test Tasks**	• Writing a test plan • Writing test cases • Development of criteria for the success of testing. • Conducting the testing and evaluation of the results • Creating test reports

| 10 | **Environmental needs** | Notepad
Laptop/Computer |
| 11 | **responsibilities** | |

Functionality and Responsibilities	Responsible
Undo	Test Engineer 1
Cut/copy	Test Engineer 2
paste	Test Engineer 3
delete	Test Engineer 4
find	Test Engineer 5
goto	Test Engineer 5

| 12 | **Staffing and Training Needs** | To perform the tasks, you need to have the following knowledge and skills:

• practical knowledge application of the notepad is needed.

• knowledge and ability to apply in practice the basic techniques of test design.

• Knowledge of various types of testing including functional and non-functional. |

		13	**Schedule**	The deadline for completion of all works and delivery of the project is 31/12/2023 by 5.00pm	
		14	**Risks and Contingencies Possible risks during testing**	• Insufficient human resources for testing the application in deadlines. • Changing the requirements for the product	
		15	**Approvals Team**	Lead Test engineer 1 Test Manager Quality Manager	
	c)	**Explain defect Management Process with suitable diagram.**			**4 M**
	Ans	 i. **Defect Prevention**-- Implementation of techniques, methodology and standard processes to reduce the risk of defects.			2 M for Diagram 2 M for Explanation

		ii. **Deliverable Baseline**-- Establishment of milestones where deliverables will be considered complete and ready for further development work. When a deliverable is base lined, any further changes are controlled. Errors in a deliverable are not considered defects until after the deliverable is base lined. iii. **Defect Discovery**-- Identification and reporting of defects for development team acknowledgment. A defect is only termed discovered when it has been documented and acknowledged as a valid defect by the development team member(s) responsible for the component(s) in error. iv. **Defect Resolution**-- Work by the development team to prioritize, schedule and fix a defect, and document the resolution. This also includes notification back to the tester to ensure that the resolution is verified. v. **Process Improvement** -- All problems are due to failure in the process involved in creating software. Defects give an opportunity to identify the problem with process used and update them. Better processes mean better product with less defect. vi. **Management Reporting** -- Analysis and reporting of defect information to assist management with risk management, process improvement and project management.	
	d)	**State and explain any four benefits of automation in testing.**	**4 M**

_____

	Ans	• **Reduces time of testing**	1 M for each benefit
		Software tests have to be repeated often during development cycles to ensure quality. Every time source code is modified software tests should be repeated. For each release of the software it may be tested on all supported operating systems and hardware configurations. Manually repeating these tests is costly and time consuming. Once created, automated tests can be run over and over again at no additional cost and they are much faster than manual tests.	
		• **Greater accuracy:** Continuous testing increases the chances of errors while done manually, but in automated testing, repetitive tests can be performed with the same precision.	
		• **Deliver the quality product**	
		The exposure and expertise of a manual tester determine how thoroughly your applications get tested. Test automation, when implemented accurately, removes your dependency on these parameters, delivering expected results every time. Yet another aspect that impacts quality is manual errors. Irrespective of how meticulous a quality engineer is, there is always a possibility of missing a few steps, entering the wrong data, or any trivial manual error – leading to poor application quality. With test automation, you can ensure that your solution will execute the steps accurately without fail and report each step without bias	
		• **Allow to run tests many time with different data**	
		They can even be run on multiple computers with different configurations. Automated software testing can look inside an application and see memory contents, data tables, file contents, and internal program states to determine if the product is behaving as expected. Test automation can easily execute thousands of different complex test cases during every test run providing coverage that is impossible with manual tests.	

		• **Save resources or requires less**	
		Manual testing, especially regression testing, can seem extremely tedious as you need to test every minor code change. Writing scripts, running them over and over again is nothing short of a nightmare. With no-code test automation, you need not write test cases or execute them manually when the codebase changes. Instead, your solution creates the test scripts which you can reuse and execute as required without additional efforts or costs.	

4.		**Attempt any THREE of the following:**	12 M
	a)	**What is boundary value analysis? Explain with suitable example.**	4 M

_____
_

	Ans		Explanation – 2 M, Example - 2 M
		• Most of the defects in software products hover around conditions and boundaries. • 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 both valid and invalid boundary values. Typically, we choose one test case from each boundary. The basic idea in boundary value testing is to select input variable values at their: 1. Minimum 2. Just below the minimum 3. Just above the minimum 4. Just below the maximum 5. Maximum 6. Just above the maximum **Example: Input Box should accept the Number 1 to 10** Here we will see the Boundary Value Test Cases	

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

b)	**Explain the Regression Testing. State when the Regression testing shall be done?**	**4 M**

_____

| | **Ans** | Regression testing a black box testing technique that consists of re-executing those tests that are impacted by the code changes.
ii. These tests should be executed as often as possible throughout the software development life cycle. Types of Regression Tests:
i. **Final Regression Tests: -** A "final regression testing" is performed to validate the build that hasn't changed for a period of time. This build is deployed or shipped to customers.
ii. **Regression Tests: -** A normal regression testing is performed to verify if the build has NOT broken any other parts of the application by the recent code changes for defect fixing or for enhancement.

When to Perform:

Regression testing should be performed whenever there is a change in the software code, configuration, or environment. For example, regression testing should be done after adding new features, fixing bugs, refactoring code, integrating components, updating libraries, or migrating to a different platform. Regression testing should also be done before releasing the software to the end-users or deploying it to the production environment. | 2 M

(for explanation related with regression testing)

1 M |
| **c)** | | **What is Test Plan? What is its need? List test planning activities.** | **4 M** |

_____

	Ans	**Test Plan:**	1 M
		i. Test plan is the project plan for the testing work to be done.	
		ii. It is not a test design *specification,* a collection of *test cases* or a set of *test procedures;* in fact, most of our test plans do not address that level of detail.	
		Need:	1 M
		i. Test Plan Ensures all Functional and Design Requirements are implemented as specified in the documentation. ii. To provide a procedure for Unit and System Testing.	
		iii. To identify the documentation process for Unit and System Testing.	
		iv. To identify the test methods for Unit and System Testing.	
		Activities:	
		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.	2 M
		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.	
	d)	**Prepare defect report for login field of email application.**	**4 M**

Ans	A defect report is a document that describes a defect, including its severity, priority, and steps to replicate the problem. A defect report's primary purpose is to help the developers quickly reproduce and fix the fault. **Defect Report Template** • **Defect id :** • **Project Name :** • **Module Name :** • **Sub Module Name :** • **Type of Defect : (wrong, missing or extra)** • **Status : (New, open, assign, fix)** • **Severity : (high, medium, low)** • **Priority : (high, medium, low)** • **Summary :** • **Description : (Steps To Reproduce)** • **Expected Result :** • **Actual Results :** • **Reported By :** • **Assign To :** • **Date & Time:** **Example: Defect Report for User Login Page** • **Defect id:** D001 • **Project Name:** Web-Site Domain Name • **Module Name:** Login • **Sub Module Name:** User Login • **Type of Defect:** Missing • **Status:** New • **Severity:** High • **Priority :** • **Summary:** User Login Page Not Opening • **Description :** 1. Enter Url / Domain Name 2. Click On the User Login Option • **Expected Result:** User login page should get open	2 M 2 M

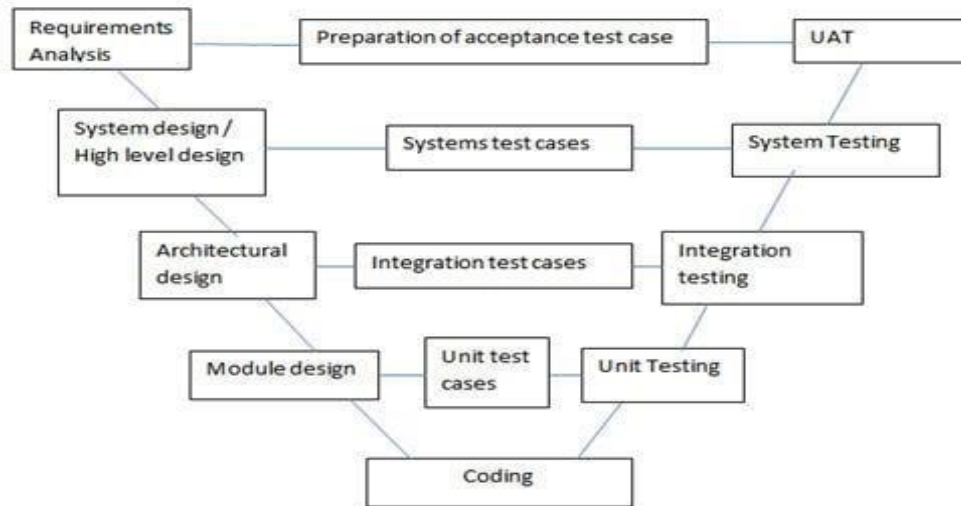|			
		• **Actual Results:** User login page does not get open • **Reported By: ABC Tester** • **Assign To: XYZ Developer** • **Date & Time: 12/28/2022**	
	e)	**State any four limitations of manual testing.**	**4 M**
	Ans	1. Manual testing is slow and costly. 2. It is very labor intensive; it takes a long time to complete tests. 3. Manual tests don't scale well. As the complexity of the software increases the complexity of the testing problem grows exponentially. This leads to an increase in total time devoted to testing as well as total cost of testing. 4. Manual testing is not consistent or repeatable. Variations in how the tests are performed as inevitable, for various reasons. One tester may approach and perform a certain test differently from another, resulting in different results on the same test, because the tests are not being performed identically. 5. Lack of training is the common problem. 6. GUI objects size difference and color combinations are not easy to find in manual testing. 7. Not suitable for large scale projects and time bound projects. 8. Batch testing is not possible, for each test execution Human user interaction is mandatory. 9. Comparing large amounts of data is impractical. 10. Processing change requests during software maintenance takes more time.	1 M for each limitation
5.		**Attempt any <u>TWO</u> of the following:**	**12 M**
	a)	**Describe V-model with labelled diagram.**	**6 M**

_____
_

| | **Ans** | • The V-model is a type of SDLC model where process executes in a sequential manner in V-shape.
 • It is also known as Verification and Validation model.
 • It is based on the association of a testing phase for each corresponding development stage.
 • Development of each step is directly associated with the testing phase.
 • The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.
 • V-Model contains Verification phases on one side of the Validation phases on the other side.
 • Verification and Validation phases are joined by coding phase in V-shape. | Diagram-2M, Explanation of Verification Phase- 2 M, Explanation of Validation Phase-2 M

 OR

 Answer with Relevant Contents |

_____
_

Verification Phase (Design Phase):

It involves static analysis technique (review) done without executing code. It is the process of evaluation of the product development phase to find whether specified requirements meet.

1. **Requirement Analysis:** This phase contains detailed communication with the customer to understand their requirements and expectations. This stage is known as Requirement Gathering.

2. **System Design:** This phase contains the system design and the complete hardware and communication setup for developing product.

3. **Architectural Design:** System design is broken down further into modules taking up different functionalities. The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood.

4. **Module Design:** In this phase the system breaks dowm into small modules. The detailed design of modules is specified, also known as Low-Level Design (LLD).

Validation (Testing Phases) :

It involves dynamic analysis technique (functional, non-functional), testing done by executing code. Validation is the process to evaluate the software after the completion of the development phase to determine whether software meets the customer expectations and requirements.

1. **Unit Testing:** Unit Test Plans are developed during module design phase. These Unit Test Plans are executed to eliminate bugs at code or unit level.

2. **Integration testing:** After completion of unit testing Integration testing is performed. In integration testing, the modules are integrated and the system is

_____

tested. Integration testing is performed on the Architecture design phase. This test verifies the communication of modules among themselves.

3. **System Testing:** System testing test the complete application with its functionality, inter dependency, and communication. It tests the functional and non-functional requirements of the developed application.

_____

–

		4. User Acceptance Testing (UAT): UAT is performed in a user environment that resembles the production environment. UAT verifies that the delivered system meets user's requirement and system is ready for use in real world.	
	b)	**Describe with one example each** **: i. Load Testing ii.** **Stress Testing**	**6 M**

_____

	Ans	**i. Load Testing**	Description of each- 2 M
		• Load Testing is a type of performance testing to check system with constantly increasing the load on the system until the time load reaches its threshold value.	
		• Here Increasing load means increasing number of concurrent users, transactions & check the behavior of the application under test.	
		• It is normally carried out underneath controlled environment to distinguish between two different systems.	
		• The main purpose of load testing is to monitor the response time and staying power of application when the system is performing well under heavy load.	
		• The successfully executed load testing is only if the specified test cases are executed without any error in allocated time.	
		• Load testing is testing the software under customer expected load.	
		• In order to perform load testing on the software you feed it all that it can handle. Operate the software with the largest possible data files.	
		• If the software operates on peripherals such as printer, or communication ports, connect as many as you can.	
		• If you are testing an internet server that can handle thousands of simultaneous connections, do it. With most software it is important for it to run over long periods.	
		• Some software should be able to run forever without being restarted. So, Time acts as an important variable. Load testing can be best applied with the help of automation tools.	
		Examples of load testing:	Any Suitable example of each – 1 M
		• Downloading a series of large files from the internet.	
		• Running multiple applications on a computer or server simultaneously.	
		• Assigning many jobs to a printer in a queue.	
		• Subjecting a server to a large amount of traffic.	
		• Writing and reading data to and from a hard disk continuously **ii. 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 behaviour 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, disk space, network request etc.)	
		• Stress testing is also called fatigue testing.	

_____

–

		• **For Example:** Word processor software running on your computer with all available memory and d^{er} space, it works fine. But if the system runs low on resources, you have a grea potential^{ure} to expect a bug. Setting the values to zero or near zero will make the softw execute a^{ure} different path as it attempts to handle the tight constraint. Ideally the softw would run without crashing or losing data.			sk

	c)	**Prepare six test cases for marketing site www.flipkart.com**	**6 M**

Ans							6 test cases of test cases: 6 M; 1 M each; any other valid test cases shall be considered

Test Case -ID	Steps	Input data	Expect ed Result	Actual Result	Status
TC-1	Type correct user name	Abc123	It Should accept user name	It accepts user name	pass
TC-2	Type correct and valid password	Co5i518	It Should accept password	It accepts password	pass
TC-3	Click on login button	-	Home page should be displayed after login and user name should be displayed on home page	Home page is displayed after login and user name should be displayed on home page	Pass
TC-4	Click on any product displayed on home page	-	User should be redirected to product specification page.	User redirected to product specification page	pass
TC-5	Click on add to cart for the product.		The product should be added to cart.	The product added to cart	Pass
TC-6	Click on go to cart button.		The total amount of all items in cart should be displayed.	The total amount of all items in cart is displayed.	pass

_____

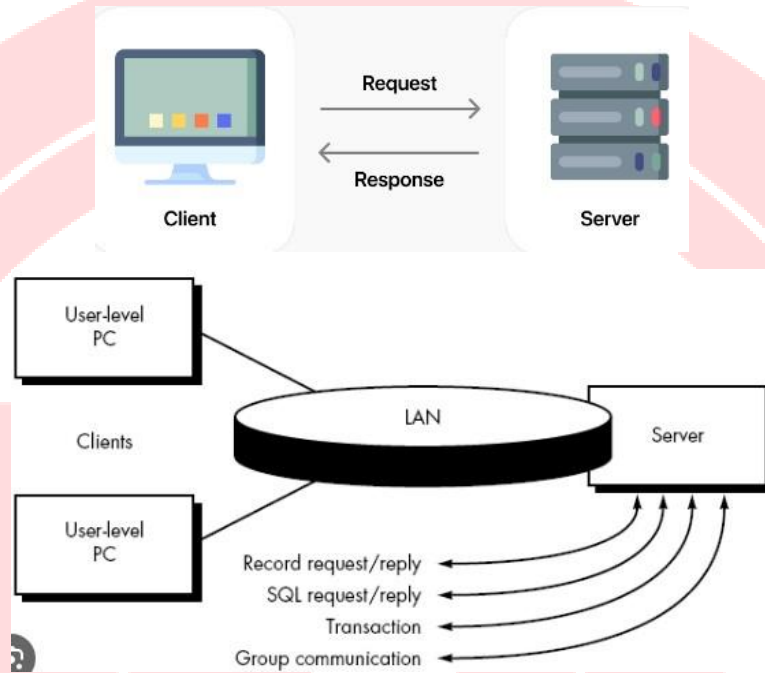|							
_		TC-7	Click on remove from cart button.	-	The item should be removed form cart.	The item is removed from the cart.	pass
		TC-8	Click on Checkout button.	-	The checkout page should be displayed with payments options.	The checkout page is displayed with payments options.	Pass
		TC-9	Make payment for the order.	-	User should get order details by message or email.	User gets order details by message or email.	Pass

6.		Attempt any **TWO** of the following:	12 M
	a)	Explain client-server testing with suitable example.	6 M

_____

Ans	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. 4. Testing approaches of client server system: • **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, whereas testing client a server simulator, and to test network both simulators are used at a time. • **Integration testing:** After successful testing of server, client and network, they are brought together to form system testing. • **Performance testing:** System performance is tested when number of clients is 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. • **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.		Description - 3 M, Any suitable example - 1 M

_____

- **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.

- **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.
- **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.

For example, the average number of users working simultaneously on a system must be quantified, since performance testing most commonly tests performance under workload stress. Testers should also determine maximum or peak user performance or how the system operates under maximum workloads. Bandwidth is another necessary bit of information, as is most users most frequent actions. Performance testing also validates and verifies other performance parameters such as reliability and scalability. Performance testing can establish that a product lives up to performance standards necessary for commercial release. It can compare two systems to determine which one performs better. Or they can use profilers to determine the programs behavior as it runs. This determines which parts of the program might cause the most trouble and it establishes thresholds of acceptable response times.

b)	**Write important six test cases for the "Login Form" of the Facebook website.**	**6 M**

Ans							6 test cases of test cases: 6 M; 1 M each; any other
	Test_ case_ no	**Test step**	**Test data**	**Expected output**	**Actual output**	**Status**	
	1	Username filed is left blank		It will display 'Enter Username'	It displays 'Enter Username'	Pass	

_____

		2	Enter invalid user name	abc	It will prompt 'couldn't find your account' message	It prompt couldn't find your account.	Pass	valid test cases shall be considered
		3	Enter valid user name and invalid password	Username- abc123 Password – 123	It will display 'Wrong password' message.	It displays 'wrong password' message.	Pass	
		4	Enter Valid username and no password	Username- abc123 Password –	It will display 'Enter password'.	It displays 'Enter password'.	Pass	

		5	Enter Valid username and Password	Username- abc123 Password – co5i22518	It will display users' account's facebook page.	It displays users account's facebook page.	Pass	
		6	Click on 'Forgotten password?'	-	It will go to Find your account page.	It goes to Find your account page.	Pass	
	c)	**Describe defect life cycle with neat diagram.**						**6 M**

	Ans		Diagram: 2 M
			Description: 4 M

NEW
OPEN
ASSIGN
REJECTED
DEFERRED
REOPENED
TEST
VERIFIED
CLOSED

OR

New
Assigned
Open
Duplicate
Rejected
Deferred
Not A Bug
Fixed
Pending Retest
Reopened
Retest
Verified
Closed
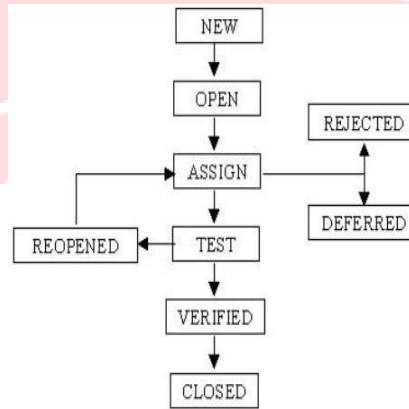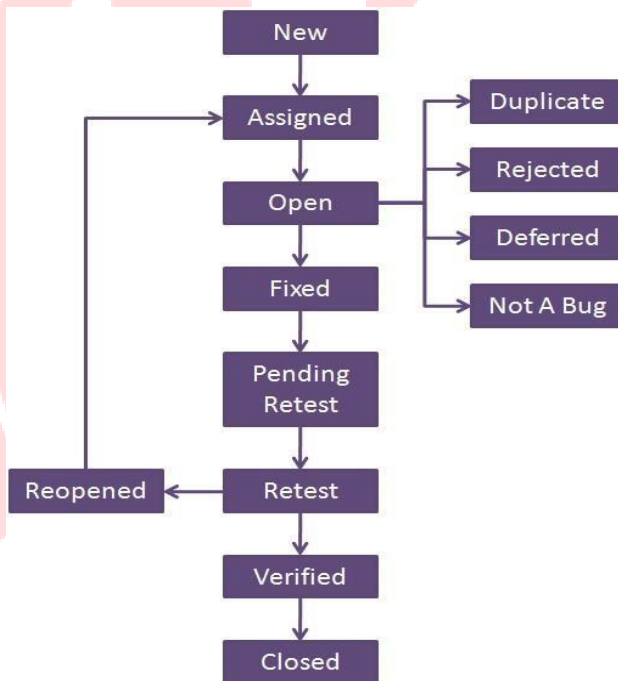
1. New: When a defect is logged and posted for the first time. It's state is given as new.

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 analysing 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 state 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 feel of the application like change of colour of some text then it is not a bug but just some change in the look of the application.

_