

SEMESTER END EXAMINATION

Subject: Software Testing (BT-CS-PE506)

SET - 2

Max. Marks: 100 Time: 3 Hours

Instructions:

1. This paper contains two sections: **Section A** and **Section B**.
 2. **Section A** is compulsory (Short Answer Questions).
 3. **Section B** contains Descriptive Questions.
 4. Assume missing data suitably if any.
-

SECTION A – Short Answer Questions

(Attempt all questions. Each question carries 2 marks.)

[10 × 2 = 20 Marks]

Q1. Differentiate between Testing and Debugging.

Answer:

- **Testing:** The process of executing software with the intent of finding defects. It is often a destructive process usually performed by testers.
- **Debugging:** The process of locating the root cause of the defect found during testing and fixing the code. It is a constructive process performed by developers.

Q2. Define the "Pesticide Paradox" in software testing.

Answer: The Pesticide Paradox states that if the same set of test cases is executed repeatedly, eventually those test cases will cease to find new defects. To overcome this, test cases must be regularly reviewed, revised, and new tests must be added.

Q3. What is the difference between Statement Coverage and Branch Coverage?

Answer:

- **Statement Coverage:** Measures the percentage of executable statements in the source code that have been executed at least once.
- **Branch Coverage:** Measures the percentage of decision outcomes (True/False edges) from control structures (like 'if', 'while') that have been executed.

Q4. What are Stubs and Drivers? Give one difference.

Answer: They are dummy modules used in integration testing.

- **Stubs:** Replace missing lower-level modules; used in Top-Down integration.

- **Drivers:** Replace missing higher-level (calling) modules; used in Bottom-Up integration.

Q5. Define Smoke Testing vs. Sanity Testing.

Answer:

- **Smoke Testing:** Initial "health check" on a new build to ensure critical functions work before deep testing begins (broad and shallow).
- **Sanity Testing:** Performed after bug fixes to ensure the fix works and didn't break related modules (narrow and deep focus).

Q6. What is the "Big Bang" approach in Integration Testing?

Answer: Big Bang integration involves combining all modules of the system simultaneously and testing them as a whole. It is suitable for small systems but risky for large systems as isolating the cause of defects is difficult.

Q7. Explain the concept of Random Testing.

Answer: Random testing is a black-box technique where test inputs are generated purely at random from the input domain without any specific pattern or guideline. It is useful for robustness testing but may miss specific boundary logic.

Q8. Who is a Test Manager? Mention one primary responsibility.

Answer: A Test Manager is the leader of the testing team responsible for the overall success of the testing effort. A primary responsibility is **resource management** (allocating testers, tools, and environments) and **reporting** status to stakeholders.

Q9. What is User Documentation Testing?

Answer: It involves verifying the manuals, user guides, installation guides, and help files provided with the software. The goal is to ensure the documentation is accurate, consistent with the software behavior, and easy to understand for the end-user.

Q10. Define "Test Adequacy Criteria".

Answer: These are the rules or standards used to determine whether enough testing has been performed. For example, "Stop testing when 100% Statement Coverage is achieved" or "Stop when 95% of high-priority test cases pass."

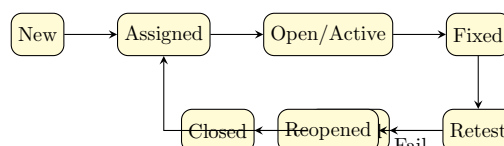
SECTION B – Descriptive Questions

Part 1: 5-Mark Questions (Attempt any 4)

(Answer should be structured and concise. Each question carries 5 marks.) [4 × 5 = 20 Marks]

Q11. Draw and explain the "Defect Life Cycle" (Bug Life Cycle).

Answer: The Defect Life Cycle represents the states a bug goes through from detection to closure.



Key States:

- **New:** Defect is logged by tester.
- **Assigned:** Lead assigns it to a developer.
- **Fixed:** Developer resolves the bug.
- **Retest:** Tester checks the specific fix.
- **Reopened:** If the fix fails, it goes back to developer.
- **Closed:** If the fix works, the bug is closed.

Q12. Briefly explain Cause-Effect Graphing technique.

Answer: **Cause-Effect Graphing** is a black-box testing technique used to analyze combinations of input conditions (causes) and their resulting system responses (effects).

- **Causes:** Input conditions (e.g., "File exists", "User is Admin").
- **Effects:** Output actions (e.g., "Delete file", "Show Error").
- **Graphing:** Uses boolean logic operators (AND, OR, NOT) to link causes to effects diagrammatically.
- **Benefit:** It helps in discovering gaps in requirements and generates a minimum set of test cases to cover maximum logic combinations, often converted into a Decision Table.

Q13. Explain Sandwich (Hybrid) Integration Testing.

Answer: Sandwich Integration is a hybrid approach combining Top-Down and Bottom-Up integration techniques.

- The system is viewed in three layers: The Target Layer (Middle), The Layer Above, and The Layer Below.
- **Approach:** The top layers are tested using Top-Down (with stubs) and bottom layers using Bottom-Up (with drivers) simultaneously, converging at the target layer.
- **Advantages:** Combines the benefits of both approaches; parallel testing is possible.
- **Disadvantages:** Complex to plan; requires both stubs and drivers.

Q14. List the standard components of a Test Case.

Answer: A well-written test case should contain the following fields:

- (a) **Test Case ID:** Unique identifier (e.g., TC_{LOGIN}001).
- (a) **Test Scenario/Objective:** What is being verified.
- (b) **Pre-conditions:** State of system before test (e.g., "User logged out").
- (c) **Test Data:** Specific inputs used (e.g., "Username: admin").
- (d) **Test Steps:** Step-by-step actions to execute.
- (e) **Expected Result:** The correct system behavior.
- (f) **Actual Result:** (Filled during execution).
- (g) **Status:** Pass/Fail.

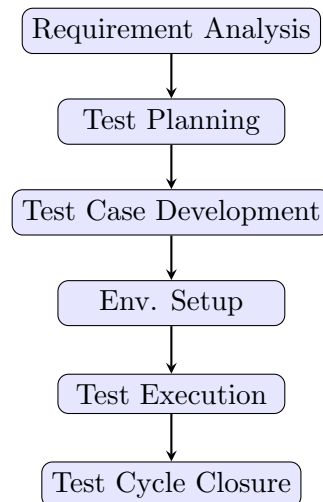
Part 2: 10-Mark Questions (Attempt any 6)

(Detailed answer required with diagrams/examples. Each question carries 10 marks.) [6 × 10 = 60 Marks]

Q15. Explain the "Software Testing Life Cycle" (STLC) with a neat flow diagram. Describe the exit criteria for the Test Execution phase.

Answer: The STLC is a sequence of specific activities performed during the testing process to ensure software quality. It runs parallel to the SDLC.

Phases of STLC:



- (a) **Requirement Analysis:** Studying requirements to identify testable items.
- (b) **Test Planning:** Estimating resources, strategy, and schedule.
- (c) **Test Case Development:** Writing test cases and scripts.
- (d) **Environment Setup:** Preparing hardware/software test beds.
- (e) **Execution:** Running tests and logging bugs.
- (f) **Closure:** Analyzing metrics and preparing closure reports.

Exit Criteria for Test Execution:

- All critical/high priority test cases are passed.
- Pass rate meets the defined threshold (e.g., 95%).
- No critical defects (Severity 1) are open.
- Regression testing is completed successfully.

Q16. Consider the following C code for finding the greatest of three numbers. Draw the Control Flow Graph (CFG) and calculate Cyclomatic Complexity.

```
1 void findGreatest(int a, int b, int c) {  
2     if (a > b) { // Node 1  
3         if (a > c) // Node 2  
4             print(a); // Node 3  
5         else  
6             print(c); // Node 4  
7     } else {  
8         if (b > c) // Node 5  
9             print(b); // Node 6
```

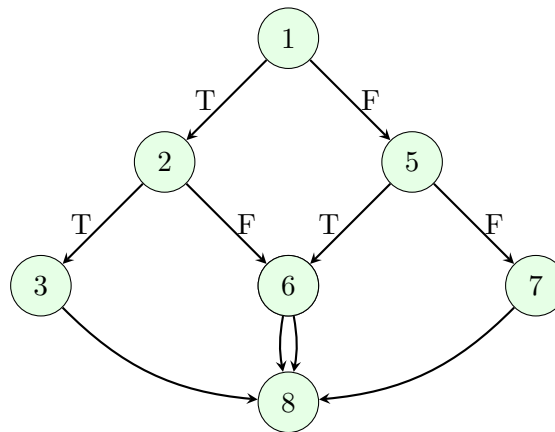
```

10         else
11             print(c);           // Node 7
12     }
13     print("Done");           // Node 8
14 }

```

Answer:

1. Control Flow Graph (CFG):



2. Cyclomatic Complexity Calculation:

- **Method 1 (Nodes - Edges):** $V(G) = E - N + 2$.
- Edges (E) = 10. Nodes (N) = 8.
- $V(G) = 10 - 8 + 2 = 4$.
- **Method 2 (Predicate Nodes):** $V(G) = P + 1$.
- Predicates (Decision points): Nodes 1, 2, 5. So, $P = 3$.
- $V(G) = 3 + 1 = 4$.

Q17. What is Decision Table Based Testing? Create a Decision Table for a login screen with fields: Username, Password, and a Rule that locks the account after 3 failed attempts.

Answer: Decision Table Testing is a black-box technique used for functions that respond to a combination of inputs. It is structured in a tabular form with Conditions (Inputs) and Actions (Outputs).

Scenario: Login Screen.

- C1: Username Valid?
- C2: Password Valid?
- C3: Failed Attempts ≥ 3 ?

Decision Table:

Conditions / Rules	R1	R2	R3	R4
C1: Username Correct?	T	T	T	F
C2: Password Correct?	T	F	F	-
C3: Attempts < 3?	-	T	F	-
Actions				
A1: Grant Access	X			
A2: Show "Invalid Pwd"		X		
A3: Show "Invalid User"				X
A4: Lock Account			X	

Explanation:

- **R1:** Valid user/pass → Access Granted.
- **R2:** Valid user, wrong pass, attempts OK → Error Message.
- **R3:** Valid user, wrong pass, max attempts reached → Lock Account.
- **R4:** Invalid user → Generic Error (Password check irrelevant).

Q18. Discuss the Organizational Structures for Testing Teams. Compare Centralized vs. Decentralized testing teams.

Answer: The structure of a testing team impacts independence and quality.

1. Centralized Testing Team:

- All testers belong to a dedicated QA department, independent of development projects. They are "loaned" to projects.
- **Pros:** High independence (unbiased), standardized processes, better resource sharing.
- **Cons:** May lack domain knowledge of specific projects, "us vs them" mentality with developers.

2. Decentralized (Project-Based) Team:

- Testers report directly to the Project Manager of the specific product.
- **Pros:** Better collaboration with developers, deep product knowledge, faster feedback.
- **Cons:** Less independence (Project Manager might pressure to skip tests to meet deadlines), inconsistent standards across company.

3. Hybrid Model: Center of Excellence sets standards, but testers sit with project squads (Agile squads). This is currently the most popular model.

Q19. What are the key challenges in Website Testing? Explain testing considerations for Security, Performance, and Compatibility of a web application.

Answer: Website testing is complex due to the distributed nature of the web and variety of user environments.

- **Compatibility Testing (Cross-Browser/Cross-Platform):**
 - The website must render correctly on Chrome, Firefox, Safari, Edge.
 - Must work on Mobile (iOS/Android) and Desktop (Windows/Mac/Linux).
 - Test for responsive design (CSS layout shifts).

- **Security Testing:**
 - **URL Manipulation:** Check if internal pages can be accessed by editing URL params.
 - **SQL Injection:** Ensure input fields sanitize data.
 - **Session Management:** Verify session timeout and cookie security (HttpOnly, Secure flags).
- **Performance Testing:**
 - **Load Testing:** Can the server handle 10,000 concurrent users?
 - **Latency:** Check page load speeds (optimizing images/scripts).

Q20. Write short notes on: (a) Usability Testing, (b) Accessibility Testing.

Answer:

(a) **Usability Testing:**

- Focuses on "How easy is it to use?".
- Testers observe real users attempting to perform tasks.
- **Metrics:** Time to complete task, error rate, user satisfaction (survey).
- **Example:** Ensuring the "Checkout" button is visible and the flow is intuitive.

(b) **Accessibility Testing (a11y):**

- Ensures the software is usable by people with disabilities (vision, hearing, motor).
- Compliance with standards like **WCAG** (Web Content Accessibility Guidelines).
- **Checks:** Screen reader compatibility (ALT text for images), keyboard navigation (no mouse required), color contrast ratios for color-blind users.