

SEMESTER END EXAMINATION

Subject: Software Testing (BT-CS-PE506)

SET - 3

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 Walkthrough and Inspection.

Answer:

- **Walkthrough:** An informal static testing technique where the author explains the code/document to peers to gather feedback. It is led by the author.
- **Inspection:** A formal, rigorous static testing technique with a defined process (moderator, reader, scribe) to detect defects based on checklists. It is led by a trained moderator.

Q2. What is "Defect Clustering"?

Answer: Defect Clustering is a testing principle based on the Pareto Principle (80/20 rule). It states that a small number of modules typically contain the majority of the defects discovered during pre-release testing.

Q3. Define Internationalization (I18n) Testing.

Answer: It is the process of verifying that the software can be adapted to various languages and regions without requiring engineering changes to the source code. It checks if the code separates data/content from logic (e.g., Unicode support).

Q4. What is "Domain Testing"?

Answer: Domain Testing is a black-box technique where the input space is analyzed to identify domains (groups of inputs) that are treated similarly by the program. It focuses on testing the logic that selects the path based on these domains.

Q5. Explain the concept of "Mutation Testing".

Answer: Mutation Testing is a white-box technique used to evaluate the quality of test cases. It involves introducing small changes (mutations) to the source code and running the test suite. If the tests fail (kill the mutant), the tests are effective; if they pass, the tests are insufficient.

Q6. What are "Entry Criteria" in a Test Plan?

Answer: Entry Criteria are the prerequisite conditions that must be met before a specific testing phase can begin. Examples include "Code must be frozen," "Smoke test passed," or "Test environment is ready."

Q7. Define "Ad-hoc Testing".

Answer: Ad-hoc testing is an informal testing type performed without any documentation, planning, or test cases. The tester relies on their intuition and experience to explore the system and find defects that formal cases might miss.

Q8. What is a "Test Incident Report"?

Answer: A Test Incident Report (or Bug Report) documents any event that occurs during testing which requires investigation. It typically includes the summary, steps to reproduce, expected vs. actual results, and severity of the issue.

Q9. Distinguish between Functional and Non-Functional Testing.

Answer:

- **Functional:** Verifies *what* the system does (features, business requirements). Example: Login, Calculate Tax.
- **Non-Functional:** Verifies *how* the system performs (quality attributes). Example: Load time, Security, Scalability.

Q10. What is "Configuration Testing"?

Answer: Configuration Testing validates the application's performance and behavior across different hardware and software configurations (e.g., different OS versions, browser types, memory sizes, or network speeds).

SECTION B – Descriptive Questions

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

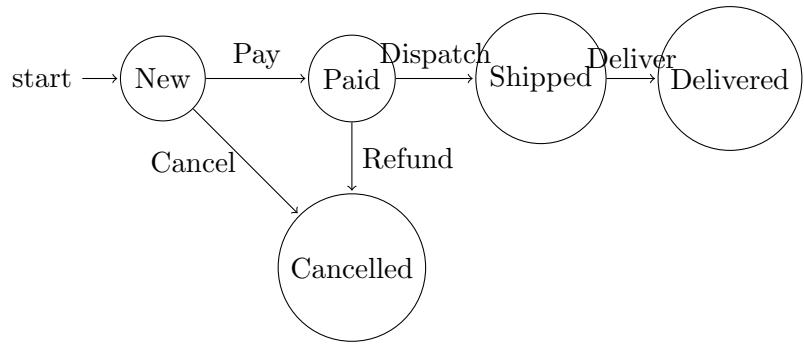
(Answer should be structured and concise. Each question carries 5 marks.) [4 × 5 = 20 Marks]

Q11. Explain State Transition Testing with a suitable diagram for an "Order Processing System".

Answer: State Transition Testing is a black-box technique used when the system's output depends on its current state and the input event.

Example: Order Processing System

- **States:** New, Paid, Shipped, Delivered, Cancelled.
- **Transitions:** Triggered by events like "Payment Success", "Dispatch", "Customer Cancel".

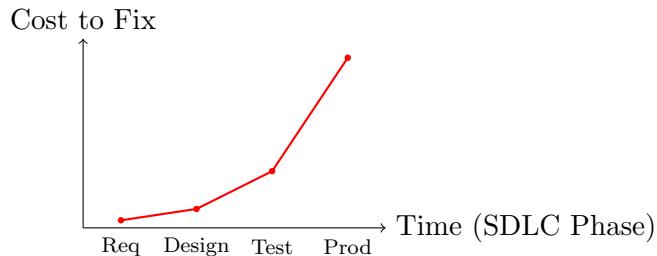


Test cases are derived to cover all valid transitions (e.g., New → Paid → Shipped) and invalid transitions (e.g., New → Delivered).

Q12. Discuss the "Cost of Quality" in software testing. Use a graph to explain the Cost of Defect detection over time.

Answer: The cost of fixing a defect increases exponentially as the software moves through the SDLC.

- **Prevention Costs:** Training, Requirements Review (Lowest cost).
- **Appraisal Costs:** Testing, Inspections.
- **Failure Costs:** Fixing bugs in Production, Customer support, Reputation loss (Highest cost).



Explanation: Fixing a bug during Requirements might cost \$10. Fixing the same bug after release might cost \$10,000 due to patches, downtime, and regression testing.

Q13. What is Data Flow Testing? Explain "Definition" and "Usage" of variables.

Answer: **Data Flow Testing** is a white-box technique that focuses on how variables are assigned values and used throughout the execution paths.

- **Definition (def):** A statement where a variable is assigned a value (e.g., 'x = 5;', 'input(y);').
- **Usage (use):** A statement where the value of a variable is accessed.
 - **P-use (Predicate use):** Used in a condition (e.g., 'if (x < 0)').
 - **C-use (Computational use):** Used in a calculation (e.g., 'z = x + y;').

The goal is to test paths from the definition of a variable to its subsequent usage (def-use chains) to ensure no variables are used before initialization or defined but never used.

Q14. Explain the difference between Test Policy and Test Strategy.

Answer:

Test Policy	Test Strategy
A high-level document describing the organization's philosophy toward testing.	A product-level document describing <i>how</i> testing will be carried out for a specific project.
Created by Senior Management.	Created by Test Manager.
Generic and applicable to all projects in the company.	Specific to the project scope and risks.
Example: "We will achieve CMMI Level 3".	Example: "We will use Selenium for Regression".

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 Acceptance Testing and its types (Alpha, Beta, Contract, Regulation) in detail.

Answer: Acceptance Testing is the final level of testing performed to determine if the software is ready for delivery. It verifies the system against business requirements.

Types of Acceptance Testing:

(a) User Acceptance Testing (UAT):

- Performed by the actual end-users to verify if the system supports their day-to-day business processes.
- Focuses on usability and workflows.

(b) Alpha Testing:

- Performed at the developer's site by internal teams (but not the developers).
- Controlled environment.

(c) Beta Testing:

- Released to a select group of external customers.
- Uncontrolled, real-world environment. Feedback is collected for the final release.

(d) Contract Acceptance Testing:

- Performed to verify if the developed software meets the criteria specified in the contract or Service Level Agreement (SLA).

(e) Regulatory/Compliance Acceptance Testing:

- Ensures software complies with government regulations (e.g., HIPAA for healthcare, GDPR for data privacy).

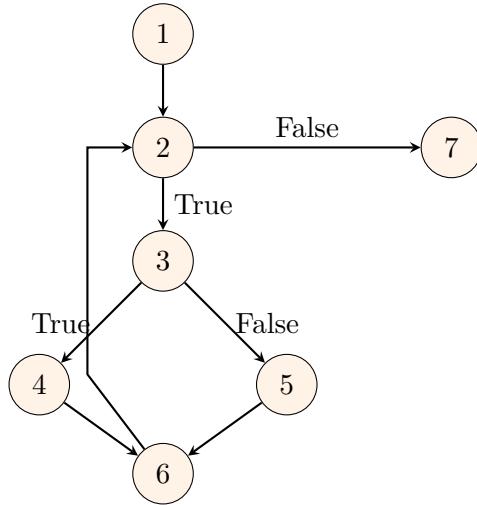
Q16. Consider the following C code snippet containing a loop. Draw the Control Flow Graph (CFG) and calculate the Cyclomatic Complexity.

```

1 void calculateSum(int n) {
2     int i = 1, sum = 0;           // Node 1
3     while (i <= n) {            // Node 2 (Condition)
4         if (i % 2 == 0)          // Node 3 (If)
5             sum = sum + i;       // Node 4
6         else
7             print(i);           // Node 5
8         i++;                   // Node 6
9     }
10    print(sum);                // Node 7
11 }
```

Answer:

1. Control Flow Graph (CFG):



2. Cyclomatic Complexity Calculation:

- **Method 1 (Regions):** Total enclosed regions + 1 (outside). Regions: 1 inside ‘if’, 1 inside ‘loop’, 1 outside. Total = 3.
- **Method 2 (Nodes/Edges):**
 - Nodes (N) = 7.
 - Edges (E) = 9 (1→2, 2→3, 2→7, 3→4, 3→5, 4→6, 5→6, 6→2).
 - $V(G) = 9 - 7 + 2 = 4$. Wait, let’s re-verify loop back.
 - Correct Edges: (1-2), (2-3), (2-7), (3-4), (3-5), (4-6), (5-6), (6-2). Total = 8 edges.
 - $V(G) = 8 - 7 + 2 = 3$.
- **Method 3 (Predicates):**
 - Predicate Nodes: Node 2 (While), Node 3 (If).
 - $V(G) = P + 1 = 2 + 1 = 3$.

Final Complexity = 3.

Q17. Compare White Box Testing strategies: Condition Coverage vs. Multiple Condition Coverage (MCDC). Provide examples.

Answer: These strategies focus on testing the boolean expressions in control structures.

(a) Condition Coverage (Basic):

- Requires that each individual boolean sub-expression evaluates to both True and False at least once.
- **Example:** ‘if (A \neq 0 OR B \neq 0)’
- Test 1: A=1, B=0 (A is T). Test 2: A=0, B=1 (B is T). Test 3: A=0, B=0 (A and B are F).
- Does not guarantee all combinations of the final result.

(b) Multiple Condition Decision Coverage (MCDC):

- A stronger criterion required for safety-critical systems (e.g., Avionics).

- Requires that every condition in a decision has taken all possible outcomes AND each condition is shown to independently affect the decision's outcome.
- **Example:** 'if (A or B)'
- We need pairs where changing **only** A changes the result, and changing **only** B changes the result.
- Test Set: (F, F) → F; (T, F) → T; (F, T) → T.

Q18. What are Test Metrics? Explain any three key metrics used to monitor test progress and quality.

Answer: Test Metrics are quantitative measures used to estimate the progress, quality, and health of the software testing effort.

(a) **Defect Density:**

- Number of confirmed defects per unit size of code (e.g., per KLOC or per Module).
- Formula: $\frac{\text{Total Defects}}{\text{Total Lines of Code (KLOC)}}$
- Indicates the quality of the module.

(b) **Test Case Execution Coverage:**

- Percentage of test cases executed out of the total planned.
- Formula: $\frac{\text{Executed Tests}}{\text{Total Tests}} \times 100$
- Indicates progress against the schedule.

(c) **Defect Fix Rate:**

- The speed at which developers are fixing reported bugs.
- Formula: $\frac{\text{Defects Fixed}}{\text{Defects Reported}} \times 100$
- Indicates the efficiency of the development team.

Q19. Discuss the "Reporting Test Results" phase. Outline the structure of a Test Summary Report.

Answer: Reporting is the communication bridge between the testing team and stakeholders. The **Test Summary Report** is generated at the end of a testing cycle.

Structure of Test Summary Report:

- **Project Overview:** Name, Version, Date.
- **Objective:** What was tested (Scope).
- **Testing Summary:**
 - Total Tests Planned: 100
 - Executed: 100
 - Passed: 90
 - Failed: 10
- **Defect Summary:**
 - Critical: 0, High: 2, Medium: 8.
 - Status of defects (Open vs Closed).
- **Variances:** Any deviation from the original plan (e.g., modules skipped).
- **Recommendation:** Go / No-Go decision for release.
- **Lessons Learned:** What went well and what didn't.

Q20. Explain Integration Test Planning. What is "Scenario Testing" in this context?

Answer: **Integration Test Planning** involves defining the strategy to combine individual units and test their interfaces.

- It identifies the sequence of integration (Top-Down, Bottom-Up, Big Bang).
- It defines the test data required to flow between modules.

Scenario Testing:

- A technique used during integration and system testing where valid business flows (scenarios) are tested end-to-end.
- Unlike checking individual features, it checks a "Story".
- **Example:** "User purchases an item."
 - (a) User Logs in (Module A).
 - (b) Search Item (Module B).
 - (c) Add to Cart (Module C).
 - (d) Payment (Module D).
 - (e) Invoice Generation (Module E).
- Integration testing ensures the data (User ID, Item ID, Amount) passes correctly from Module A → B → C → D.