

Enrollment No./Seat No.:

GUJARAT TECHNOLOGICAL UNIVERSITY
MBA - SEMESTER - III EXAMINATION - WINTER 2025

Subject Code: MB03096031

Date: 19-12-2025

Subject Name: Fundamentals of Software Quality Assurance

Time: 10:30 AM TO 01:30 PM

Total Marks: 70

Instructions

- 1. Attempt all questions.**
- 2. Make suitable assumptions wherever necessary.**
- 3. Figures to the right indicate full marks.**
- 4. USE of SIMPLE calculators AND non-programmable scientific calculators are permitted.**

	Marks
Q.1 ALL 7 Questions must be Compulsory.	14
(a) What is Software Quality Assurance (SQA)?	
(b) What is Prevention Cost?	
(c) What is benchmarking in software quality?	
(d) Define Test Planning.	
(e) Define Boundary Value Testing (BVT).	
(f) Define Path Testing.	
(g) Define Software Verification.	
Q.2 (a) Distinguish between fact-based quality and perceived quality with suitable examples from software products.	07
(b) Differentiate between testing and debugging with suitable examples. Why is it important to distinguish them?	07
OR	
(b) Short note on Requirement Traceability Matrix (RTM)	07
Q.3 (a) What is Equivalence Class Partitioning (ECP)? Explain how inputs are divided into groups for testing and also describe the difference between Traditional ECP and Improved ECP.	07
(b) What are Program Graphs, DD-Paths, and Control Flow Graphs? Explain with example	07
OR	
(a) What do you mean by Software Verification and Validation? Explain the difference between them with suitable examples.	07
(b) Describe some common verification methods like inspections, reviews, and walkthroughs with examples	07

- Q.4 (a)** Define quality from different perspectives: **07**
- Product-based
 - Manufacturing-based
 - Customer-based
- (b)** Give a brief account of the historical context and definitions of software testing. How has testing evolved over time? **07**

OR

- (a)** Write the difference between ECP and BVT with examples. **07**
- (b)** What is a Verification Workbench? Explain its purpose in the software development process. **07**

Q.5 Case Study: Unit Testing of an Online Banking Transaction System

An online banking application allows customers to transfer money between accounts, with strict validation and verification processes at each step. The system requires users to log in using their customer ID and password. Once logged in, they can initiate a fund transfer by entering the recipient's account number, transfer amount, and selecting the transaction type (NEFT, RTGS, or IMPS). The valid transfer amount must be between ₹100 and ₹2,00,000, and the system ensures that the user's account balance is sufficient before proceeding. If any validation fails, an appropriate error message is displayed. To ensure the reliability and correctness of this critical module, the development team applies multiple test design techniques during unit testing, including Boundary Value Testing (BVT), Equivalence Class Partitioning (ECP), Decision Table Testing, Path Testing, and Data Flow Testing.

- (a)** Apply Boundary Value Testing (BVT) to the transaction amount input, which ranges between ₹100 and ₹2,00,000. Identify Normal, Robust, and Worst-Case boundary test cases, and also include any special values that need to be tested, explaining why they are important for validating the system's robustness. **07**
- (b)** Using Equivalence Class Partitioning (ECP), identify valid and invalid input classes for both the customer ID (which must contain exactly 8 digits) and password (which must contain 6–12 alphanumeric characters, including at least one special symbol). Show how Traditional and Improved ECP can be used to minimize the number of test cases while maintaining test coverage, and mention the edge test cases that would be most critical. **07**

OR

- (a)** Using the principles of Path Testing, identify all possible execution paths in the transaction validation module where conditions such as invalid login, insufficient balance, and unsupported transaction type may occur. Explain how achieving 100% branch coverage ensures that all decision outcomes are tested and discuss why this is crucial for financial software systems like online banking. **07**
- (b)** Apply Data Flow Testing to the same transaction module and identify the define-use pairs for key variables such as balance, amount, and transaction_status. Explain how anomalies like "defined but not used" or "used before definition" can lead to software defects, and describe how data flow analysis helps prevent such issues during unit testing. **07**
