CBCS SCHEME

USN | | | | | | | | | | BIS714B

Seventh Semester B.E./B.Tech. Degree Examination, Dec.2025/Jan.2026
Software Quality Assurance

Time: 3 hrs. Max. Marks: 100

Note: 1. Answer any FIVE full questions, choosing ONE full question from each module.
2. M : Marks , L: Bloom's level , C: Course outcomes.

		Module – 1	M	L	C
Q.1	a.	Define SQA. Summarize the main characteristics of the environment for which software quality assurance (SQA) methods were developed.	10	L2	CO3
	b.	Explain the main objectives of software quality assurance activities in a software project.	06	L2	CO3
	c.	Define the following terms : (i) Software (ii) Software errors (iii) Software faults (iv) Software failures	04	L1	CO3
		OR			
Q.2	a.	List nine causes of software errors and discuss any 3 in detail with suitable example.	10	L1 L2	CO3
	b.	Describe the key categories and associated factors in McCall's software quality model using an illustrative model tree diagram.	10	L2	CO3
		Module – 2			
Q.3	a.	List components of software project life cycle. Discuss review and software maintenance components in detail.	10	L2	CO2
	b.	List and describe software quality assurance infrastructure components that help in error prevention and process improvement.	10	L2	CO2
		OR			
Q.4	a.	List and explain the objectives of proposal draft review.	10	L2	CO2
	b.	Discuss the factors that affect the extent of the contract review.	04	L2	CO2
	c.	Describe the recommended avenues for implementing a major contract review.	06	L2	CO2
		Module – 3			
Q.5	a.	Describe the prototyping model with the help of a neat diagram. Also discuss its advantages and disadvantages.	08	L2	CO1
	b.	Identify and discuss three types of data used in the application model for measuring SQA defect removal effectiveness and cost.	06	L2	CO1
	c.	Define software testing and describe the key terms related to the formal operative characteristics of testing.	06	L1 L2	CO1

		OR			
Q.6	a.	Discuss the different stages of the top-down testing approach with help of a neat diagram.	08	L2	CO1
	b.	Explain black box and white box testing concepts with suitable examples. Also list their advantages and disadvantages.	12	L1 L2	CO1
		Module – 4			
Q.7	a.	Interpret the main components of project progress control and describe how each contribute to effective project management.	10	L2	CO4
	b.	Briefly discuss how computerized tools are used to control both risk management activities and project schedules during project execution.	10	L2	CO4
		OR			
Q.8	a.	Discuss the various categories of software development process metrics with suitable example.	08	L2	CO4
	b.	With neat diagram discuss the 4 stage process of defining software quality metrics.	08	L2	CO4
	c.	Define the concept of Help Desk (HD) services and describe three types of HD quality metrics with example.	04	L1 L2	CO4
		Module – 5			
Q.9	a.	Describe the steps involved in ISO 9000 – 3 certification process with neat diagram.	10	L2	CO5
	b.	With neat diagram discuss the structure of ISO/IEC 15504 process assessment model.	10	L2	CO5
		OR			
Q.10	a.	Using Fishbone diagram illustrate and explain the architecture of IEEE/EIA Std 12207 software life cycle processes.	10	L2	CO5
	b.	State and explain the underlying concepts of verification and validation according to IEEE standard 1012 – 1998.	10	L2	CO5

* * * * *

Q1 (a) Define SQA. Summarize the main characteristics of the environment in which SQA methods were developed. (10 Marks)

Definition of Software Quality Assurance (2 Marks)

Software Quality Assurance (SQA) is a planned and systematic set of activities implemented throughout the software development life cycle to ensure that software processes and products conform to specified requirements, standards, and procedures, and that the final product meets the desired quality level.

Characteristics of the Environment in which SQA Methods were Developed (8 Marks)

SQA methods evolved due to the following environmental characteristics:

1. Increasing Software Complexity
   Modern software systems are large, distributed, and highly complex, making manual error detection difficult and necessitating structured quality practices.
2. High Cost of Software Failures
   Software failures can result in financial loss, safety risks, legal liabilities, and loss of organizational reputation, especially in safety-critical systems.
3. Process-Oriented Development
   Quality cannot be ensured only by final testing; it must be built into each phase of development through defined processes.
4. Customer-Driven Quality Requirements
   Users expect reliability, usability, maintainability, efficiency, and security, increasing the need for systematic quality assurance.
5. Reduced Development Time (Time-to-Market Pressure)
   Rapid release cycles demand early defect detection and prevention to avoid costly rework.
6. Need for Standardization
   Adoption of international standards such as ISO, IEEE, and CMMI ensures consistency and repeatability in quality practices.
7. Advancement in Tools and Technology
   Use of automated testing tools, CASE tools, and metrics-based monitoring supports effective SQA implementation.
8. Global and Distributed Development
   Development across different locations requires formal documentation, reviews, audits, and communication standards.

Q1 (b) Explain the main objectives of software quality assurance activities in a software project. (6 Marks)

The main objectives of Software Quality Assurance activities are:

1. Ensuring Compliance with Standards
   To verify that software development follows organizational, national, and international standards.
2. Defect Prevention
   To identify and eliminate defects at early stages rather than detecting them after development.

3. Process Improvement
   To continuously evaluate and improve development and maintenance processes.
4. Improvement of Product Quality
   To ensure the software meets functional and non-functional requirements.
5. Risk Reduction
   To minimize technical, schedule, and cost risks in the project.
6. Customer Satisfaction
   To deliver reliable, maintainable, and high-quality software products.

Q1 (c) Define the following terms:
(4 Marks)
(i) Software Error (1 Mark)
A software error is a human mistake made during software development that results in incorrect logic or understanding.
(ii) Software Fault (1 Mark)
A software fault is a defect in the code or design caused by an error.
(iii) Software Failure (1 Mark)
A software failure occurs when the software does not perform its intended function under specified conditions.
(iv) Software Defect (1 Mark)
A software defect is any deviation of the software from its specified requirements or expected behavior.

Q2 (a) List nine causes of software errors and discuss any three in detail with suitable examples.
(10 Marks)
List of Nine Causes of Software Errors (5 Marks)

I. Incomplete or unclear requirements
II. Poor communication with stakeholders
III. Time pressure and unrealistic schedules
IV. Lack of domain knowledge
V. Poor design methodology
VI. Human mistakes (programmer oversight)
VII. Inadequate testing
VIII. Use of improper development tools
IX. Frequent changes in requirements

Discuss Any Three Causes in Detail with Examples (5 Marks)
1. Incomplete or Unclear Requirements
If requirements are not clearly defined, developers may misunderstand system functionality, leading to incorrect implementation.

Example: If a banking system requirement does not clearly specify withdrawal limits, developers may implement incorrect transaction rules, causing financial errors.

2. Time Pressure and Unrealistic Schedules

Strict deadlines force developers to skip proper design, testing, and documentation, increasing error probability.

Example: Rushed delivery of an e-commerce website may result in payment gateway failures due to insufficient testing.

3. Inadequate Testing

When testing is insufficient, defects remain undetected and appear during actual usage.

Example: A mobile app released without performance testing may crash frequently when multiple users access it simultaneously.

Q2 (b) Describe the key categories and associated factors in McCall's software quality model using an illustrative model tree diagram. (10 Marks)

Introduction to McCall's Software Quality Model (2 Marks)

McCall's software quality model defines software quality in terms of 11 quality factors, organized into three major categories, providing a structured way to evaluate software quality.

Three Key Categories of McCall's Model (3 Marks)

1. Product Operation – How well the software operates for users
2. Product Revision – How easily the software can be modified
3. Product Transition – How easily the software adapts to new environments

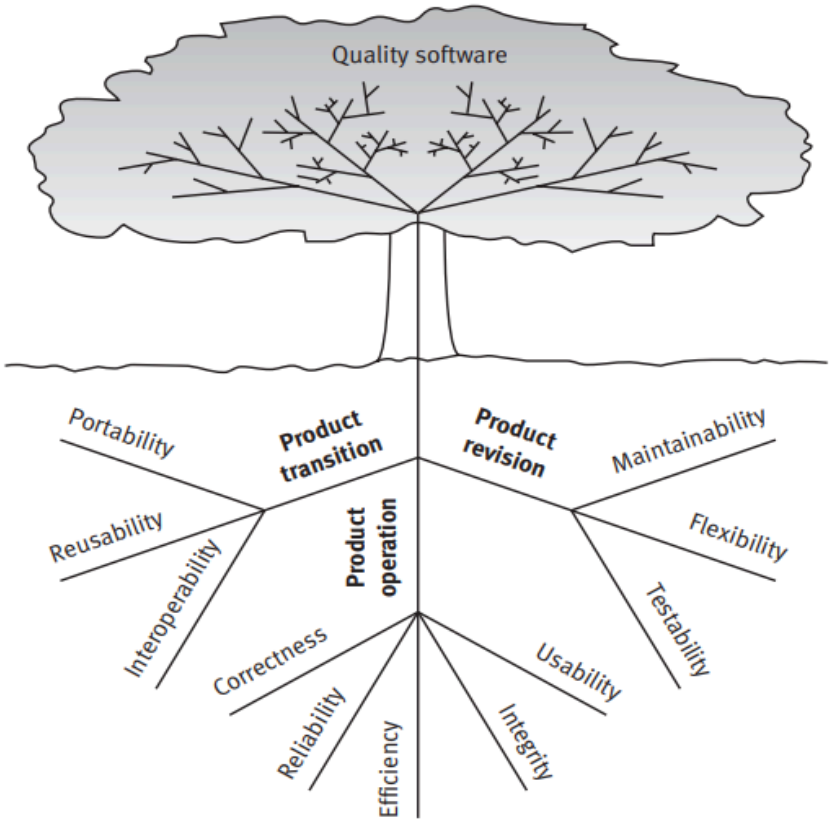
Quality Factors under Each Category (3 Marks)

1. Product Operation Factors
   - ☐ Correctness
   - ☐ Reliability
   - ☐ Efficiency
   - ☐ Integrity
   - ☐ Usability
2. Product Revision Factors
   - ☐ Maintainability
   - ☐ Flexibility
   - ☐ Testability
3. Product Transition Factors
   - ☐ Portability
   - ☐ Reusability
   - ☐ Interoperability

Associated Factors Explanation (2 Marks)

- ☐ Correctness: Degree to which software meets requirements
- ☐ Reliability: Ability to function without failure

- ☐ Efficiency: Optimal use of system resources
- ☐ Integrity: Protection against unauthorized access
- ☐ Usability: Ease of learning and use
- ☐ Maintainability: Ease of fixing defects
- ☐ Flexibility: Ease of modification
- ☐ Testability: Ease of testing
- ☐ Portability: Ability to run on different platforms
- ☐ Reusability: Ability to reuse components
- ☐ Interoperability: Ability to interact with other systems



Sure 👍
Here is Q3 rewritten fully in detail, in descriptive VTU exam-writing style, with expanded explanations, sub-points, and examples — suitable for full 20/20 marks.

Q3. Software Project Life Cycle & SQA Infrastructure

Q3 (a) List components of software project life cycle. Discuss review and software maintenance components in detail.
(10 Marks)

Components of Software Project Life Cycle (4 Marks)

The software project life cycle (SPLC) is a structured sequence of phases followed during software development. The main components are:

1. Feasibility Study
   Analysis of technical, economic, and operational feasibility of the project.
2. Requirements Analysis & Specification
   Identification and documentation of functional and non-functional requirements.
3. System Design
   Creation of architectural, database, and interface designs.
4. Coding / Implementation
   Conversion of design into executable programs.
5. Testing
   Verification and validation of software to detect defects.
6. Deployment
   Installation and delivery of software to the user.
7. Review
   Formal evaluation of project artifacts for defect detection.
8. Maintenance
   Post-delivery modification of software.

(Any 6–8 components: 4 marks)

Review Component – Detailed Explanation (3 Marks)

Definition:

A software review is a systematic, formal examination of software documents, design, code, and test artifacts to detect defects early in the development life cycle.

Objectives of Review:

1. To identify defects at early stages
2. To ensure compliance with standards
3. To improve overall software quality
4. To reduce development and maintenance cost
5. To enhance reliability and correctness

Types of Software Reviews:

1. Informal Review
   - Casual checking by peers
   - No formal procedure
2. Walkthrough
   - Author presents work to peers

- Feedback is collected

3. Formal Technical Review (FTR)
   - Structured review process
   - Moderated by trained personnel
4. Inspection
   - Most formal review
   - Uses checklists, defined roles, and defect logs

Advantages of Reviews:
- Early defect detection
- Reduced rework
- Improved documentation quality
- Better team understanding
- Increased software reliability

Example: A design review detects ambiguous database relationships before coding, preventing major redesign effort later.

Software Maintenance Component – Detailed Explanation (3 Marks)

Definition:

Software maintenance refers to modification of software after delivery to correct faults, improve performance, or adapt to changes in environment and user needs.

Objectives of Software Maintenance:
1. Correct defects
2. Improve performance
3. Add new features
4. Adapt to new hardware/software platforms
5. Enhance reliability and security

Types of Software Maintenance:

1. Corrective Maintenance
   - Fixes errors found during operation

Example: Fixing login failure bug

---

2. Adaptive Maintenance
   - Adapting software to new environment

Example: Migrating application from Windows to Linux

---

3. Perfective Maintenance
   - Improves performance or adds new features

Example: Adding online payment option

4. Preventive Maintenance

- Improves maintainability and reduces future risk

Example: Code refactoring and restructuring

Importance of Software Maintenance:

- Extends software life
- Improves customer satisfaction
- Reduces system downtime
- Enhances reliability

Q3 (b) List and describe software quality assurance infrastructure components that help in error prevention and process improvement. (10 Marks)

Introduction to SQA Infrastructure (2 Marks)

SQA infrastructure refers to the organizational framework, procedures, tools, and methodologies that support systematic implementation of quality assurance practices in a software organization.

Its main goals are:

- Error prevention
- Continuous process improvement
- Quality enhancement

Major SQA Infrastructure Components (8 Marks)

1. SQA Organizational Framework

Defines roles, responsibilities, and authority of SQA personnel.

Functions:

- Planning quality activities
- Conducting audits and reviews
- Reporting quality status

2. Documentation Control System

Ensures proper creation, updating, storage, and retrieval of documents.

Benefits:

- Maintains traceability
- Avoids outdated documents
- Supports audits

3. Standards, Procedures & Guidelines

Defines coding standards, review checklists, testing procedures.

Advantages:

- Uniformity
- Improved quality
- Reduced ambiguity

4. Training & Certification Programs

Provides training on quality standards, tools, and processes.

Benefits:

- Enhances developer skills
- Improves defect prevention
- Builds quality culture

5. Metrics and Measurement System
Collects and analyzes data such as defect density, productivity, and reliability.
Examples:
- Defects per KLOC
- Mean time between failures (MTBF)

6. Formal Review & Audit Mechanisms
Periodic assessment of processes and deliverables.
Functions:
- Detect defects early
- Ensure compliance
- Suggest improvements

7. Configuration Management System (SCM)
Controls changes and manages software versions.
Functions:
- Version control
- Change tracking
- Build management

8. Corrective and Preventive Action (CAPA) System
Identifies root causes and prevents recurrence of defects.
9. SQA Tools & Automation Support (Extra)
Tools for testing, bug tracking, continuous integration, and reporting.
10. Continuous Process Improvement Framework (Extra)
Uses feedback, metrics, and audits to continuously enhance processes.
Advantages of SQA Infrastructure:
- Reduces defect rates
- Improves development efficiency
- Enhances customer satisfaction
- Ensures predictable quality

Q4 (a) Explain contract review process. Discuss its objectives, stages, and importance. (10 Marks)

Introduction: Contract review is a systematic and formal evaluation of customer requirements and contractual obligations before accepting and signing a software development contract. The objective is to ensure that all technical, financial, legal, and managerial aspects are clearly understood, feasible, and achievable.

Objectives of Contract Review: The main objectives of contract review are:

1. To ensure complete understanding of customer requirements
2. To verify technical feasibility of the project
3. To assess cost, schedule, and resource feasibility
4. To identify risks and constraints
5. To ensure clarity in deliverables, milestones, and acceptance criteria
6. To avoid future disputes and misunderstandings

Stages of Contract Review Process: The major stages involved in the contract review process are:

1. Requirement Analysis Stage
   In this stage, the customer's functional and non-functional requirements are carefully analyzed to ensure clarity and completeness.
2. Technical Feasibility Analysis
   The availability of suitable technologies, tools, infrastructure, and skilled manpower is evaluated.
3. Resource and Cost Estimation
   Required manpower, project duration, cost, and hardware/software resources are estimated.
4. Risk Assessment
   Potential technical, financial, and managerial risks are identified, and mitigation strategies are planned.
5. Negotiation and Clarification
   Discussions are held with the client regarding pricing, delivery schedules, penalties, and warranty clauses.
6. Final Approval and Contract Signing
   Senior management reviews all aspects and approves the contract before signing.

Importance of Contract Review: Contract review is important because it prevents misunderstandings, reduces project risks, ensures realistic commitments, avoids cost and schedule overruns, and improves customer satisfaction.

Q4 (b) Explain proposal review process. Discuss its objectives, stages, and significance. (10 Marks)

Introduction: Proposal review is the systematic evaluation of a project proposal before it is submitted to the customer. It ensures that the proposal is technically correct, financially feasible, competitive, and compliant with customer requirements.

Objectives of Proposal Review: The main objectives of proposal review are:

1. To ensure technical correctness and feasibility
2. To verify the accuracy and competitiveness of cost estimates
3. To ensure compliance with client requirements
4. To identify risks and weaknesses in the proposal

5. To improve clarity, completeness, and presentation quality

Stages of Proposal Review Process: The important stages in proposal review are:

1. Understanding Customer Requirements
   The request for proposal (RFP) is studied to understand project scope, constraints, and expectations.
2. Development of Technical Solution
   Appropriate system architecture, tools, and development strategy are designed.
3. Cost and Schedule Estimation
   Estimates are prepared for manpower, time, and budget.
4. Risk Analysis
   Technical, financial, and operational risks are identified, and mitigation plans are developed.
5. Internal Review and Management Approval
   Senior technical experts and managers review the proposal for accuracy and feasibility.
6. Final Editing and Submission
   The proposal is finalized, formatted, and submitted to the customer.

Significance of Proposal Review: Proposal review increases the chances of winning projects, improves proposal quality, reduces execution risks, ensures customer satisfaction, and enhances the organization's reputation.

Q5 (a) Explain the Prototyping Model. Discuss its advantages and limitations. (10 Marks)

Introduction: The Prototyping Model is a software development approach in which a working model of the system, called a prototype, is developed early in the development process. This prototype helps users and developers understand system requirements more clearly before building the final product.

Working of Prototyping Model

1. Requirement Gathering: Initial requirements are collected from users. These requirements may be incomplete or unclear.
2. Quick Design: A preliminary design is prepared focusing on major system functionalities.
3. Prototype Development: A simplified version of the software is built based on quick design.
4. User Evaluation: Users interact with the prototype and provide feedback.
5. Refinement of Requirements: Based on feedback, requirements are modified and improved.
6. Final System Development: The actual system is developed using the refined requirements.

Advantages of Prototyping Model

1. Clear understanding of user requirements
2. Early detection of defects and missing functionality

3. Improved user involvement and satisfaction
4. Reduced risk of project failure
5. Better communication between users and developers

Limitations of Prototyping Model

1. Increases development cost and time
2. Poor documentation may occur
3. Frequent changes can cause scope creep
4. Not suitable for large and complex systems
5. Prototype may be mistaken as final product

Q5 (b) Explain software quality metrics. Discuss different categories of software metrics with examples. (10 Marks)

Introduction: Software quality metrics are quantitative measures used to evaluate the quality, performance, and productivity of software development processes and products. Metrics help in monitoring progress, detecting problems, and improving quality.

Objectives of Software Metrics

1. Measure software quality
2. Monitor development progress
3. Improve productivity
4. Reduce defects
5. Support management decision-making

Categories of Software Metrics

1. Product Metrics
   Measure characteristics of the software product.
   Examples:
   - Lines of Code (LOC)
   - Defect density
   - Reliability
   - Maintainability index
2. Process Metrics
   Measure effectiveness of the development process.
   Examples:
   - Defect removal efficiency
   - Review efficiency
   - Development cycle time

3. Project Metrics
   Measure project management performance.
   Examples:
   - Cost variance
   - Schedule variance
   - Productivity
   - Resource utilization
4. Quality Metrics
   Measure quality-related attributes.
   Examples:
   - Mean time to failure (MTTF)
   - Mean time between failures (MTBF)
   - Customer-reported defects

Benefits of Software Metrics

1. Improved quality control
2. Early detection of problems
3. Better project planning
4. Performance evaluation
5. Continuous improvement

Q6 (a) Explain formal technical review (FTR). Discuss its objectives, participants, and review process. (10 Marks)

Introduction: Formal Technical Review (FTR) is a structured group review of software documents, design, or code with the objective of detecting defects early and improving quality.

Objectives of FTR

1. Detect defects early
2. Ensure compliance with standards
3. Improve software quality
4. Promote knowledge sharing
5. Reduce rework

Participants in FTR

1. Moderator – Conducts and controls the review meeting
2. Author – Presents the work product
3. Reviewers – Examine documents and identify defects

4. Recorder – Documents defects and review outcomes

FTR Process Steps

1. Planning
2. Overview meeting
3. Individual preparation
4. Review meeting
5. Rework
6. Follow-up

Advantages of FTR:

1. Early error detection
2. Improved reliability
3. Reduced maintenance cost
4. Improved documentation quality
5. Knowledge transfer

Q6 (b) Explain software testing strategies. Discuss different testing levels and techniques. (10 Marks)

Introduction: Software testing is the process of executing software to detect defects and ensure that it meets specified requirements.

Objectives of Software Testing:

1. Detect errors
2. Verify functionality
3. Validate requirements
4. Improve quality
5. Reduce failures

Levels of Software Testing

1. Unit Testing: Testing individual modules or functions.
2. Integration Testing: Testing interaction between integrated modules.
3. System Testing: Testing the complete integrated system.
4. Acceptance Testing: Testing by end users to validate requirements.

Testing Techniques

1. Black Box Testing: Focuses on functionality without knowing internal structure.
2. White Box Testing: Focuses on internal logic, control flow, and code paths.

3. Grey Box Testing: Combination of black and white box testing.

Testing Strategies are as follows.

1. Top-down testing
2. Bottom-up testing
3. Regression testing
4. Performance testing
5. Security testing

Q7(a) Explain Software Configuration Management (SCM). Discuss its objectives, activities and benefits. (10 Marks)

Introduction: Software Configuration Management (SCM) is a systematic and disciplined process for identifying, organizing, controlling, and tracking changes in software throughout the software development life cycle. SCM ensures that software products remain consistent, correct, traceable, and manageable, even when multiple developers work simultaneously. It helps in maintaining the integrity of software by managing different versions, changes, and documentation in a controlled manner.

Objectives of SCM: The main objectives of SCM are to control software changes, maintain version consistency, ensure product integrity, track configuration status, improve coordination among team members, minimize errors caused by uncontrolled changes, and support systematic software evolution.

Major SCM Activities

1. Configuration Identification: This involves identifying and defining configuration items (CIs) such as source code files, design documents, test cases, requirement specifications, user manuals, and executable programs. Each CI is uniquely labeled and stored so that it can be tracked and managed.

2. Version Control: Version control manages different versions of configuration items. It ensures that only authorized changes are made, prevents accidental overwriting of code, supports rollback to previous versions, and enables parallel development.

3. Change Control: Change control is a formal process for handling change requests. Each request is analyzed, evaluated, approved or rejected by a Change Control Board (CCB), implemented, tested, and documented to avoid uncontrolled modifications.

4. Configuration Status Accounting: This activity records and reports the status of configuration items, versions, approved changes, and release history. It helps management track project progress and quality.

5. Configuration Audits: Configuration audits verify whether software conforms to its documented requirements, standards, and approved changes. It ensures completeness, correctness, and compliance.

Benefits of SCM: SCM improves software quality, reduces errors, enhances team coordination, ensures traceability, supports maintenance, controls project complexity, and provides reliable version management.

Q7(b) Explain software change management process. (10 Marks)

Introduction: Software change management is a structured process for controlling, evaluating, approving, and implementing changes in software systems. It ensures that all modifications are properly analyzed, documented, tested, and released in a controlled manner, minimizing risks and maintaining system stability.

Steps in Software Change Management Process

1. Change Request Submission: Users or developers submit formal change requests specifying the required modification.

2. Change Analysis: The request is analyzed to determine its technical feasibility and necessity.

3. Impact Analysis: The effect of the change on cost, schedule, performance, reliability, and other modules is evaluated.

4. Approval or Rejection: The Change Control Board (CCB) decides whether the change should be approved or rejected.

5. Change Implementation: Approved changes are implemented in the software.

6. Testing and Validation: Modified software is tested to ensure correctness and to avoid side effects.

7. Documentation and Release: All changes are documented, version updated, and released.

Advantages of Change Management: It reduces project risk, prevents scope creep, improves traceability, enhances software quality, and ensures system stability.

Q8(a) Explain software quality audits. Discuss its objectives, types and benefits. (10 Marks)

Introduction: Software quality audit is a systematic, independent, and documented examination to verify whether software processes and products comply with planned arrangements, standards, procedures, and contractual requirements. It ensures consistency, correctness, and continuous improvement.

Objectives of Software Quality Audit: The objectives are to ensure compliance with standards, detect deviations, improve development processes, enhance product quality, and ensure adherence to organizational and international quality norms.

Types of Software Quality Audits

1. Internal Audit: Conducted within the organization to evaluate internal processes and compliance.

2. External Audit: Performed by independent third-party agencies or certification bodies.

3. Process Audit: Evaluates whether development processes are correctly followed.

4. Product Audit: Examines the final software product for correctness and quality.

Benefits of Software Quality Audit

Audits improve product quality, reduce defects, ensure compliance, enhance reliability, build customer confidence, and support continuous improvement.

Q8(b) Explain software risk management. Discuss risk identification, analysis and mitigation strategies. (10 Marks)

Introduction: Software risk management is a systematic process of identifying, analyzing, prioritizing, and controlling risks that may negatively impact a software project. It helps in minimizing uncertainties and improving project success probability.

Risk Identification: Risks are identified by analyzing project environment, requirements, technology, schedule, resources, and management practices. Common risks include technical risks, cost risks, schedule risks, personnel risks, and operational risks.

Risk Analysis: Risk analysis involves estimating the probability of occurrence and the impact of each risk. Risks are then prioritized based on their severity.

Risk Mitigation Strategies

1. Risk Avoidance – Changing project plan to eliminate risk
2. Risk Reduction – Taking actions to reduce risk probability or impact
3. Risk Transfer – Transferring risk through outsourcing or insurance
4. Risk Acceptance – Accepting risk when mitigation cost is high

Benefits of Risk Management: Risk management reduces project failures, improves planning, enhances decision making, increases reliability, and ensures timely project completion.

Q9(a) Differentiate between verification and validation. Explain their importance. (10 Marks)

Difference Between Verification and Validation

Verification	Validation
Checks whether software is built correctly	Checks whether right software is built
Static process	Dynamic process
No execution required	Execution required
Reviews, inspections, walkthroughs	Testing, acceptance testing

Importance of Verification: Verification ensures early defect detection, improves product quality, reduces rework cost, and ensures compliance with standards.

Importance of Validation: Validation ensures customer satisfaction, confirms functional correctness, enhances reliability, and verifies that user needs are met.

Q9(b) Explain software documentation. Discuss its types and importance. (10 Marks)

Introduction: Software documentation refers to written materials that describe software requirements, design, implementation, testing, operation, and maintenance. It acts as a communication bridge between developers, users, testers, and maintainers.

Types of Software Documentation

1. System Documentation – Describes system architecture, design, and functionality
2. User Documentation – Provides operating instructions for end users
3. Technical Documentation – Contains detailed program logic and design details
4. Maintenance Documentation – Supports debugging and system upgrades

Importance of Documentation: Documentation improves understanding, supports maintenance, enables knowledge transfer, facilitates training, ensures consistency, and helps in quality audits.

Q10(a) Explain software maintenance. Discuss its types and challenges. (10 Marks)

Introduction: Software maintenance involves modifying and updating software after delivery to correct defects, improve performance, add new features, and adapt to environmental changes. Maintenance consumes the largest portion of software lifecycle cost.

Types of Software Maintenance

1. Corrective Maintenance – Fixing errors detected after delivery
2. Adaptive Maintenance – Modifying software to suit new environments
3. Perfective Maintenance – Enhancing performance or functionality
4. Preventive Maintenance – Improving maintainability to prevent future problems

Challenges of Software Maintenance

Challenges include poor documentation, high system complexity, lack of skilled manpower, high cost, difficulty in understanding legacy systems, and frequent requirement changes.

Q10(b) Explain continuous process improvement in SQA. (10 Marks)

Introduction: Continuous process improvement in SQA refers to ongoing evaluation and enhancement of software development processes to achieve higher quality, efficiency, and customer satisfaction.

Steps in Continuous Process Improvement

1. Measurement – Collect data using metrics
2. Analysis – Identify weaknesses and root causes
3. Improvement Planning – Develop corrective strategies
4. Implementation – Apply improvements
5. Feedback & Monitoring – Evaluate effectiveness and refine processes

Benefits of Continuous Process Improvement: It reduces defects, improves productivity, enhances quality, lowers cost, ensures customer satisfaction, and promotes organizational excellence.