SYSTEM MCQs COLLECTION

NSCT Prep

Free MCQ Practice for NSCT Test Preparation

Software Engineering

1440 Multiple Choice Questions

nsctprep.dev

This dataset is created and compiled by Muhammad Abdullah Awais

Easy Questions

480 questions

Q1. What does software engineering primarily deal with?

A. Systematic software development
B. Configuring computer network infrastructure
C. Managing enterprise database systems
D. Building and maintaining hardware systems

Answer: A

Q2. Who is considered the first person to use the term 'software engineering'?

A. Alan Turing
B. Edsger Dijkstra
C. Dennis Ritchie
D. Margaret Hamilton

Answer: D

Q3. Which of the following is NOT a characteristic of good software?

A. Dependability
B. Complexity
C. Maintainability
D. Efficiency

Answer: B

Q4. What is the main goal of software engineering?

A. To incorporate the latest emerging technology
B. To minimize the overall development team size
C. To produce quality software within budget and schedule
D. To write source code as quickly as possible

Answer: C

Q5. Software engineering is a branch of which field?

A. Computer science and engineering
B. Electrical engineering and circuits
C. Mechanical engineering and design
D. Civil engineering and structures

Answer: A

Q6. What does the term 'software crisis' refer to?

A. Rapidly increasing hardware component costs
B. Insufficient programming language features
C. Problems in developing large-scale software
D. A critical shortage of trained programmers

Answer: C

Q7. Which of the following is a software product?

A. Operating System
B. Mechanical keyboard
C. Monitor display unit
D. Laser printer device

Answer: A

Q8. What is a software process?

A. A standalone computer program in the development
B. A relational database system within the project scope
C. A set of activities for software development
D. A physical hardware component during the software

Answer: C

Q9. Which attribute is essential for software quality?

A. Large file size
B. Complex algorithms
C. Reliability
D. Maximum features

Answer: C

Q10. What is the purpose of software documentation?

A. To increase overall project file size
B. To help understand and maintain the software
C. To make onboarding harder for new hires
D. To intentionally slow down the development

Answer: B

Q11. What is the Waterfall model?

A. A circular iterative development process
B. A random unstructured coding process
C. An agile methodology with sprint cycles
D. A linear sequential software development approach

Answer: D

Q12. Which phase comes first in the Waterfall model?

A. Implementation phase
B. Testing the system
C. Requirements analysis
D. Deployment to users

Answer: C

Q13. What is an iterative process model?

A. A model that omits all testing phases
B. A model that executes only a single run
C. A hardware-focused design model only
D. A model that repeats development cycles

Answer: D

Q14. What does the 'V-Model' emphasize?

A. Only source code implementation tasks
B. Hardware testing procedures and checks
C. Only requirements documentation writing
D. Verification and validation at each development stage

Answer: D

Q15. In which model are requirements expected to be fully known at the start?

A. Spiral
B. Prototype
C. Waterfall
D. Agile

Answer: C

Q16. What is a software development life cycle (SDLC)?

A. The process of planning, creating, testing, and deploying software
B. A physical computing hardware component unit in all development efforts
C. A specific type of programming language syntax for all project stakeholders
D. A specialized category of database system type within the given constraints

Answer: A

Q17. Which model is also known as the 'Classic Life Cycle'?

A. Waterfall
B. RAD to achi
C. Spiral by
D. Agile and

Answer: A

Q18. What is prototyping in software development?

A. Building a preliminary version to explore requirements
B. Delivering the final production product in the engineering
C. Performing hardware acceptance testing as part of the methodology
D. Removing outdated legacy source code for effective project outcomes

Answer: A

Q19. Which process model involves risk analysis in each iteration?

A. RAD
B. Spiral
C. V-Model
D. Waterfall

Answer: B

Q20. What is the main disadvantage of the Waterfall model?

A. Spending too much time on testing activities
B. Being excessively flexible and unstructured
C. Moving too fast through development phases
D. Difficult to accommodate changes after a phase is completed

Answer: D

Q21. What is Agile software development?

A. A hardware design and manufacturing method as defined by standards
B. An iterative approach emphasizing flexibility and customer collaboration
C. A documentation-only waterfall approach for all project stakeholders
D. A strict sequential step-by-step approach over the entire lifecycle

Answer: B

Q22. Which of the following is an Agile methodology?

A. Scrum
B. CMMI
C. V-Model
D. Waterfall

Answer: A

Q23. What is a 'Sprint' in Scrum?

A. A formal documentation writing phase only
B. A time-boxed iteration typically lasting 2-4 weeks
C. A physical running exercise for team health
D. A specific category of software defect type

Answer: B

Q24. Who is the Product Owner in Scrum?

A. The quality assurance tester on the project in all development efforts
B. The person responsible for maximizing the value of the product
C. The database administrator managing storage by the project team members
D. The chief executive officer of the company within the given constraints

Answer: B

Q25. What is the Agile Manifesto?

A. A declaration of four core values and twelve principles for Agile development
B. A programming language specification document to achieve project objectives
C. A legally binding contractual document only and related components
D. A hardware installation and setup manual in the engineering discipline

Answer: A

Q26. How many values does the Agile Manifesto contain?

A. 2
B. 8
C. 4
D. 6

Answer: C

Q27. What does Scrum emphasize?

A. Hardware design and manufacturing work
B. Individual solo development practices only
C. Self-organizing teams and iterative progress
D. Producing extensive formal documentation

Answer: C

Q28. What is a 'User Story' in Agile?

A. A fictional novel written about end users for effective project outcomes
B. A comprehensive automated test case only in the development process
C. A formal detailed software bug report only as part of the methodology
D. A short description of a feature from the user's perspective

Answer: D

Q29. What is a Daily Standup meeting?

A. A brief daily meeting where team members share progress and blockers
B. An extensive annual strategic planning event for the development team
C. A comprehensive weekly team meeting session during the software lifecycle
D. A detailed monthly project review session within the project scope

Answer: A

Q30. Which Agile value prioritizes 'working software' over what?

A. Individual interactions
B. Responding to change
C. Customer collaboration
D. Comprehensive documentation

Answer: D

Q31. What is requirements engineering?

A. Performing software testing and validation in the development process
B. Designing hardware system architectures during the software lifecycle
C. Writing application source code for features as part of the methodology
D. The process of defining, documenting, and maintaining software requirements

Answer: D

Q32. What is a functional requirement?

A. A constraint on the system design choices
B. A measurable performance goal for the system
C. A requirement that describes what the system should do
D. A detailed hardware specification document

Answer: C

Q33. What is a non-functional requirement?

A. A requirement that specifies quality attributes like performance or security
B. A requirement that is impossible to implement for the development team
C. A broken or flawed software requirement only within the project scope
D. A specification for hardware system components throughout the project

Answer: A

Q34. What is requirements elicitation?

A. The process of gathering requirements from stakeholders
B. Performing systematic software testing runs
C. Writing application source code for modules
D. Deploying software to production servers

Answer: A

Q35. Which of the following is a requirements elicitation technique?

A. Interview sessions
B. Debugging errors
C. Coding source logic
D. Code compilation

Answer: A

Q36. What is a Software Requirements Specification (SRS)?

A. The application source code implementation
B. A comprehensive software test plan document
C. A complete end-user operational and help manual
D. A document describing all requirements of the system

Answer: D

Q37. Who are stakeholders in requirements engineering?

A. Only the programmers writing the source code
B. Only the paying customers of the software product
C. Only the project managers overseeing the work
D. Anyone who has an interest in or is affected by the system

Answer: D

Q38. What is the difference between user requirements and system requirements?

A. They are fundamentally the same concept entirely in the system context
B. User requirements are high-level; system requirements are detailed technical specifications
C. System requirements are always much simpler overall during implementation
D. User requirements are far more technically detailed for quality purposes

Answer: B

Q39. What is requirements validation?

A. Implementing requirements directly into code by the organization
B. Initially writing new requirements documents in practice typically
C. Checking that requirements are correct, complete, and consistent
D. Permanently deleting unnecessary requirements as a standard approach

Answer: C

Q40. What is a use case?

A. A description of how a user interacts with the system to achieve a goal
B. A category of reported software defect type across all phases
C. A specific automated test case scenario only at every stage
D. A step in the deployment process workflow in a systematic way

Answer: A

Q41. What is software project management?

A. Planning, organizing, and controlling software development activities
B. Designing hardware system architectures throughout the project
C. Writing application source code and modules within the project scope
D. Performing software testing and debugging for the development team

Answer: A

Q42. What is a Gantt chart?

A. A bar chart showing project schedule and task dependencies
B. A relational database schema design diagram
C. A specific type of source code format only
D. A specialized software testing automation tool

Answer: A

Q43. What does WBS stand for?

A. Work Based Standards
B. Web Browser Software
C. Work Breakdown Structure
D. Wireless Broadcasting System

Answer: C

Q44. What is a project milestone?

A. An established coding standard and guideline
B. A significant point or event in a project timeline
C. A physical unit of measured distance only
D. A specific category of software defect type

Answer: B

Q45. What is the critical path in project management?

A. The longest sequence of dependent tasks determining the minimum project duration
B. The path with the fewest total listed tasks for quality purposes
C. The shortest possible project path for tasks during implementation
D. The set of easiest tasks in the project plan in the system context

Answer: A

Q46. What is project scope?

A. The total number of developers on the team as a standard approach
B. The total size of the source code base only in practice typically
C. The defined boundaries of what the project will and will not deliver
D. Only the financial budget for the project by the organization

Answer: C

Q47. What is a deliverable in project management?

A. A tangible or intangible output produced as a result of project work
B. A scheduled meeting agenda and minutes record in a systematic way
C. A food delivery order to the project team at every stage
D. A formal software defect bug report document across all phases

Answer: A

Q48. What is project estimation?

A. Measuring physical hardware performance speeds
B. Predicting the effort, cost, and duration needed for a project
C. Simply counting total lines of application code
D. Making random guesses without any real basis

Answer: B

Q49. What is resource allocation?

A. Deleting unnecessary project data and files
B. Assigning available resources to project tasks
C. Writing comprehensive project documentation
D. Purchasing new computing hardware equipment

Answer: B

Q50. What are the three constraints of project management?

A. Design, Build, Ship within the system boundary
B. Code, Test, Deploy for the project goals
C. Scope, Time, Cost (the triple constraint)
D. Plan, Do, Check according to best practices

Answer: C

Q51. What is software design?

A. Writing detailed end-user operation manual docs as a standard approach
B. Deploying software to production server systems in practice typically
C. Writing comprehensive test case scenario suites for quality purposes
D. The process of defining the architecture, components, and interfaces of a system

Answer: D

Q52. What is modularity in software design?

A. Using only one file for all the source code
B. Dividing a system into separate, independent modules
C. Writing one single large monolithic program
D. Avoiding the use of functions in the code

Answer: B

Q53. What is cohesion in software design?

A. Unnecessary duplication of source code and logic
B. Multiple modules depending on each other heavily
C. A specific category of software defect and bug
D. The degree to which elements within a module belong together

Answer: D

Q54. What is coupling in software design?

A. A high degree of internal module cohesion only
B. A specific type of data structure or element
C. The degree of interdependence between modules
D. A programming language built-in feature type

Answer: C

Q55. Which is preferred: high cohesion or low cohesion?

A. No cohesion at all
B. Both are equally good
C. Low cohesion is preferred
D. High cohesion is preferred

Answer: D

Q56. Which is preferred: tight coupling or loose coupling?

A. Loose coupling is preferred
B. No coupling at all works
C. Tight coupling is preferred
D. Both are equally good

Answer: A

Q57. What is a design pattern?

A. A specific coding style and convention set
B. A user interface visual color scheme choice
C. A reusable solution to a commonly occurring design problem
D. A comprehensive software testing method type

Answer: C

Q58. What is abstraction in design?

A. A specific type of runtime software error type
B. Adding increasingly more implementation detail
C. Removing all application source code and files
D. Hiding complex details and showing only essential features

Answer: D

Q59. What is information hiding?

A. Concealing internal design decisions within a module from other modules
B. Permanently deleting stored information across all phases
C. Hiding files on the operating system disk at every stage
D. Encrypting sensitive data during transit by the organization

Answer: A

Q60. What is the difference between high-level and low-level design?

A. They are fundamentally the same thing exactly in a systematic way
B. High-level defines architecture; low-level defines detailed component design
C. Low-level design is always more important overall for the project goals
D. High-level design is done after coding is finished according to best practices

Answer: B

Q61. What is software architecture?

A. The high-level structure of a software system defining its components and their relationships
B. A specific type of programming language syntax in the engineering discipline
C. A comprehensive software testing framework type for effective project outcomes
D. The actual application source code itself only and related components to achieve project objectives

Answer: A

Q62. What is a client-server architecture?

A. A standalone desktop application design only during the software lifecycle
B. An architecture where clients request services from a central server
C. A decentralized peer-to-peer network system in the development process
D. A single standalone computer system by itself as part of the methodology

Answer: B

Q63. What is a layered architecture?

A. An architecture organized into horizontal layers, each providing services to the layer above
B. A completely flat system structure layout design within the project scope
C. A random unorganized structure for the code throughout the project in the system context
D. A circular system structure design layout type for the development team

Answer: A

Q64. What is a monolithic architecture?

A. A client-server communication system setup only in practice typically
B. A single-tiered application where all components are tightly integrated
C. A fully distributed computing system setup design during implementation
D. A microservices-based architecture approach type for quality purposes

Answer: B

Q65. What is the purpose of software architecture?

A. To write application source code much faster as a standard approach
B. To provide a blueprint for the system and enable communication among stakeholders
C. To completely eliminate the software testing phase by the organization
D. To reduce the overall project development team at every stage

Answer: B

Q66. What is a three-tier architecture?

A. A three-story office building design layout across all phases
B. An architecture with presentation, business logic, and data tiers
C. Three different programming languages used for the project goals
D. A development team of three programmers in a systematic way

Answer: B

Q67. What is a repository architecture?

A. A network infrastructure architecture type only by the development process
B. A Git source code version control repository according to best practices
C. An architecture where components share data through a central data store
D. An operating system file storage system only within the system boundary

Answer: C

Q68. What is a pipe-and-filter architecture?

A. An access security management system type only as defined by standards
B. A physical plumbing infrastructure system design and its related activities
C. A network traffic filtering system setup only over the entire lifecycle
D. An architecture where data flows through a sequence of processing components

Answer: D

Q69. What is an event-driven architecture?

A. An architecture where components communicate through events
B. A sequential processing architecture style only
C. A scheduling calendar management system only
D. A batch data processing system design only

Answer: A

Q70. What is a distributed system architecture?

A. An architecture where components run on multiple networked computers
B. A standalone desktop application design approach within the given constraints
C. A single standalone computer hardware unit only for all project stakeholders
D. A local desktop application installation setup in all development efforts

Answer: A

Q71. What does UI stand for?

A. User Information
B. User Interface
C. Universal Integration
D. Unified Input

Answer: B

Q72. What does UX stand for?

A. User Experience
B. Universal Exchange
C. Unified Expression
D. User Extension

Answer: A

Q73. What is usability?

A. The ease with which users can learn and use a software system
B. Application processing execution speed rating
C. System memory consumption and usage metrics
D. Internal source code logic complexity metrics

Answer: A

Q74. What is a wireframe?

A. A basic visual guide showing the layout and structure of a web page or app
B. A type of network cable connector wire type during the software lifecycle
C. A general-purpose programming language tool set within the project scope
D. A specialized software testing tool suite only for the development team

Answer: A

Q75. What is responsive design?

A. Websites that load extremely fast for users throughout the project
B. A design that responds to voice command input in the system context
C. A type of programming language paradigm type during implementation
D. Design that adapts the layout to different screen sizes and devices

Answer: D

Q76. What is a mockup in UI design?

A. A relational database schema design model type in practice typically
B. A specific automated test case scenario setup for quality purposes
C. A source code implementation template file set as a standard approach
D. A static visual representation of how the final product will look

Answer: D

Q77. What is navigation in UI design?

A. The system that allows users to move between different sections of an application
B. A physical GPS location navigation system tool by the organization
C. Network routing and traffic configuration setup at every stage
D. Operating system file directory browsing tool across all phases

Answer: A

Q78. What is consistency in UI design?

A. Using random and varying page layouts designs in a systematic way
B. Making every page look distinctly different each for the project goals
C. Changing interface colors very frequently always according to best practices
D. Maintaining uniform design elements and behaviors throughout the interface

Answer: D

Q79. What is a call-to-action (CTA) button?

A. A phone call button for customer support line
B. A window close or dismiss button on the page
C. A button designed to prompt users to take a specific action
D. A help documentation access button on the page

Answer: C

Q80. What is accessibility in UI design?

A. Database access control and permissions setup
B. Access control to application source code files
C. Designing software so it can be used by people with disabilities
D. Internet access and connectivity service plans

Answer: C

Q81. What is coding in software development?

A. Performing software testing and debugging work
B. Writing comprehensive project documentation work
C. Assembling physical hardware components together
D. The process of writing source code to implement software design

Answer: D

Q82. What is a coding standard?

A. A comprehensive software testing standard only
B. A hardware specification and standard set only
C. A specific type of source code module and file
D. A set of guidelines for writing consistent and readable code

Answer: D

Q83. What is code review?

A. Compiling source code into executable files by the project team members
B. Writing new feature source code logic only in all development efforts
C. Permanently deleting source code files only within the given constraints
D. Systematic examination of source code by peers to find defects

Answer: D

Q84. What is refactoring?

A. Permanently deleting old source code and files
B. Completely rewriting everything from scratch again
C. Adding entirely new features and functionality
D. Restructuring existing code without changing its external behavior

Answer: D

Q85. What is version control?

A. Managing changes to source code over time
B. Controlling hardware component versions only
C. A type of software testing methodology only
D. Controlling software versions in a retail store

Answer: A

Q86. What is debugging?

A. Deploying software to production server systems
B. Intentionally adding bugs to the source code
C. Writing brand new application source code only
D. The process of finding and fixing defects in code

Answer: D

Q87. What is an IDE?

A. Integrated Development Environment — a software tool for writing and debugging code
B. An internet web hosting service provider only and related components
C. A standard network communication protocol type in the engineering discipline
D. A relational database management system only to achieve project objectives

Answer: A

Q88. What is source code?

A. Human-readable instructions written in a programming language
B. Binary data stored on disk drives and memory
C. Low-level machine instruction code and binary
D. Hardware specifications and schematics diagrams

Answer: A

Q89. What is a build in software development?

A. A physical construction building project plan
B. The process of converting source code into executable software
C. A comprehensive design document and artifact
D. A specific type of software test execution run

Answer: B

Q90. What is technical documentation?

A. End user reviews and feedback comment forms as part of the methodology
B. Sales reports and revenue projections summary in the development process
C. Written descriptions of code structure, APIs, and system behavior
D. Marketing and promotional campaign materials for effective project outcomes

Answer: C

Q91. What is software testing?

A. Designing software system architecture diagrams as defined by standards
B. The process of evaluating software to find defects and verify it meets requirements
C. Writing application source code for new features and its related activities
D. Deploying software to production server systems over the entire lifecycle

Answer: B

Q92. What is a test case?

A. A set of conditions and expected results for testing a specific aspect of software
B. A specific type of source code module and library in all development efforts
C. A suitcase for carrying test documents and files within the given constraints
D. A legal court case and proceedings in a courtroom for all project stakeholders

Answer: A

Q93. What is unit testing?

A. Testing graphical user interface screens and flows
B. Testing physical hardware units and device parts
C. Testing the entire integrated system all at once
D. Testing individual components or functions in isolation

Answer: D

Q94. What is integration testing?

A. Testing documentation to achieve project objectives
B. Testing in isolation and related components
C. Testing the interaction between integrated components
D. Unit testing by the project team members

Answer: C

Q95. What is a bug in software?

A. A recognized software design pattern and approach
B. A new feature request from project stakeholders
C. A defect or error in software that causes incorrect behavior
D. An insect found inside a computer case housing

Answer: C

Q96. What is black-box testing?

A. Testing with a completely black display screen view
B. Testing physical hardware system components only
C. Testing software without knowledge of its internal code structure
D. Testing in a completely dark room with no light

Answer: C

Q97. What is white-box testing?

A. Testing with a completely white display screen view
B. Testing software with knowledge of its internal code structure
C. Testing comprehensive project documentation items
D. Testing written on white paper documents and sheets

Answer: B

Q98. What is regression testing?

A. Going backwards in the development lifecycle flow in the engineering discipline
B. Testing the software for the very first time ever for effective project outcomes
C. Testing the user registration feature flow only as part of the methodology
D. Re-testing software after changes to ensure existing functionality still works

Answer: D

Q99. What is the purpose of a test plan?

A. To outline the testing strategy, objectives, and schedule
B. To design the user interface screen layout pages
C. To deploy software to production server systems
D. To write application source code for new modules

Answer: A

Q100. What is acceptance testing?

A. Testing to determine if the software meets user requirements and is ready for delivery
B. Accepting all reported software bugs as features in the development process
C. Testing performed by internal developers only ever within the project scope
D. Testing document formatting and acceptance rules during the software lifecycle

Answer: A

Q101. What is software maintenance?

A. Writing brand new software from scratch entirely in the development process
B. Maintaining physical hardware equipment and parts during the software lifecycle
C. Installing software on new workstations and PCs within the project scope
D. Modifying software after delivery to correct faults, improve performance, or adapt to changes

Answer: D

Q102. What is corrective maintenance?

A. Improving overall system performance speed rates
B. Fixing defects and bugs discovered after deployment
C. Adapting software to new operating environments
D. Adding new features and functionality to the app

Answer: B

Q103. What is adaptive maintenance?

A. Adding brand new features to the product
B. Modifying software to work in a changed environment
C. Fixing bugs found after release for the development team
D. Improving performance and speed throughout the project

Answer: B

Q104. What is perfective maintenance?

A. Adapting software to new operating platforms only for quality purposes
B. Making software absolutely perfect and flawless in the system context
C. Fixing only critical security bugs and flaws only during implementation
D. Enhancing software functionality and performance based on user feedback

Answer: D

Q105. What is preventive maintenance?

A. Preventing all use of the software product system in practice typically
B. Making changes to prevent future problems and improve maintainability
C. Preventing all user access to the entire system by the organization
D. Preventing only software bugs from occurring only as a standard approach

Answer: B

Q106. What percentage of software cost is typically spent on maintenance?

A. 60-80%
B. 25% acr
C. 10% at
D. 5% in

Answer: A

Q107. What is a software patch?

A. A physical hardware fix and repair procedure doc
B. A completely new software release version number
C. A piece of code designed to fix or update existing software
D. A fabric patch for clothing and textile repairs

Answer: C

Q108. What is legacy software?

A. Brand new recently developed software application
B. Open source freely available software components
C. Older software that is still in use but may be difficult to maintain
D. Mobile application software for phones and tablets

Answer: C

Q109. What is software evolution?

A. Darwin's biological theory of species evolution for the project goals
B. Permanently deleting outdated software and files according to best practices
C. Physical hardware evolution over many years time within the system boundary
D. The process of continuously changing and adapting software over its lifecycle

Answer: D

Q110. What is a software release?

A. Permanently deleting the software product files
B. A press release about the software product launch
C. A version of software made available to users
D. A specific type of software defect or bug found

Answer: C

Q111. What does QA stand for in software?

A. Quantitative Assessment
B. Query Analysis
C. Quick Application
D. Quality Assurance

Answer: D

Q112. What is software quality?

A. The overall speed of software development activities
B. The total number of features available in the app
C. The degree to which software meets requirements and user expectations
D. The total length of all source code lines and files

Answer: C

Q113. What is the difference between QA and testing?

A. The two concepts are fundamentally the same exact thing across all phases
B. QA focuses on preventing defects through processes; testing focuses on finding defects
C. Testing is always much broader than QA practices are in a systematic way
D. QA only involves software testing activities and nothing else for the project goals

Answer: B

Q114. What is a software standard?

A. A physical flag or banner displayed for events according to best practices
B. A physical computing hardware component and part by the development process
C. A specific type of application source code module within the system boundary
D. A documented agreement containing rules and guidelines for software development

Answer: D

Q115. What is a code inspection?

A. A formal review of source code by peers to find defects
B. Running the compiled application directly for users
C. Compiling the source code into binary executables
D. Looking at a computer display screen only visually

Answer: A

Q116. What is a software defect?

A. An intended software feature or product capability and its related activities
B. A scheduled software update or patch file release over the entire lifecycle
C. A recognized software design pattern and approach as defined by standards
D. A flaw in software that causes it to produce incorrect or unexpected results

Answer: D

Q117. What is the purpose of quality control?

A. Checking that deliverables meet quality standards
B. Controlling the overall project scope boundary
C. Controlling physical hardware components and parts
D. Controlling the overall project team size numbers

Answer: A

Q118. What is a review in software QA?

A. A systematic examination of software artifacts to find issues
B. A movie review and rating evaluation by critics
C. An employee performance review and appraisal form
D. A consumer product rating and evaluation review

Answer: A

Q119. What is ISO 9001?

A. An international standard for quality management systems
B. A specialized software testing tool suite and program
C. A general-purpose programming language tool and set
D. A specific type of application source code file only

Answer: A

Q120. What is the cost of quality?

A. The total cost of ensuring quality plus the cost of poor quality
B. Only the prevention costs of quality effort spent
C. Only the rework and bug fixing cost totals only
D. Only the software testing costs and effort totals

Answer: A

Q121. What are software metrics?

A. A comprehensive software testing method type only for effective project outcomes
B. Quantitative measures used to assess software quality, process, and productivity
C. A specific type of application source code file type in the engineering discipline
D. Physical measurements of hardware component parts to achieve project objectives

Answer: B

Q122. What is Lines of Code (LOC)?

A. A poem or literary piece of creative written work as part of the methodology
B. A metric that measures software size by counting the number of lines in source code
C. A specialized debugging and tracing tool and suite during the software lifecycle
D. A specific type of software coding error and defect in the development process

Answer: B

Q123. What is a product metric?

A. A process metric for development workflow tracking
B. A metric used for measuring sales revenue totals
C. A measure of some attribute of the software product itself
D. A hardware performance metric for device benchmarks

Answer: C

Q124. What is a process metric?

A. A network performance metric for bandwidth speed
B. A physical hardware component metric measurement
C. A measure of the software product itself directly
D. A measure of some attribute of the software development process

Answer: D

Q125. What is productivity in software metrics?

A. The total number of meetings held per week time
B. Working overtime hours every single week constantly
C. The ratio of output (software produced) to input (resources used)
D. The typing speed of individual developer teammates

Answer: C

Q126. What is defect rate?

A. A metric for software testing execution speed rate
B. Speed of creating new defects per work hour only
C. The number of defects found per unit of software or time
D. A specific type of reported software bug and defect

Answer: C

Q127. What does KLOC stand for?

A. King Lines of Code within the
B. Known Lines of Code throughout
C. Thousands of Lines of Code
D. Key Lines of Code for the

Answer: C

Q128. What is code complexity?

A. The total number of declared variable names only for quality purposes
B. The total length of all source code lines and files during implementation
C. How hard source code is to initially write out in the system context
D. A metric measuring how difficult code is to understand, test, and maintain

Answer: D

Q129. What is mean time between failures (MTBF)?

A. The average time between system failures, indicating reliability
B. The maximum allowed software testing time duration
C. The average time to repair a system from failure
D. The minimum testing frequency and interval schedule

Answer: A

Q130. What is mean time to repair (MTTR)?

A. The average time between system failures and outages
B. The maximum time allowed for running test suites
C. The average time it takes to fix a failure and restore the system
D. The minimum time to respond to a user web request

Answer: C

Q131. What is Software Configuration Management (SCM)?

A. A creational object-oriented design pattern type
B. The process of tracking and controlling changes to software
C. A source code implementation coding technique only
D. A comprehensive software testing method type only

Answer: B

Q132. What is version control?

A. A software architecture and design method approach in the engineering discipline
B. Controlling video game software versions and builds and related components
C. A type of software testing methodology and approach to achieve project objectives
D. A system that records changes to files over time so specific versions can be recalled

Answer: D

Q133. What is a baseline in configuration management?

A. A specific line of application source code and logic as part of the methodology
B. A starting position in a sports competition event for effective project outcomes
C. A software testing benchmark result point and score in the development process
D. A formally reviewed and agreed-upon specification serving as the basis for further development

Answer: D

Q134. What is a configuration item (CI)?

A. Any artifact placed under configuration management control
B. A user story in the product backlog items listing
C. A piece of physical hardware equipment and component
D. A specific automated test case scenario and script

Answer: A

Q135. What is Git?

A. A programming language within the project
B. A testing tool for the development team
C. A distributed version control system
D. A command during the software lifecycle

Answer: C

Q136. What is a commit in version control?

A. A personal promise or commitment to other people
B. A production deployment to end users and clients
C. A recorded change or set of changes to the repository
D. A test execution run and its result log report

Answer: C

Q137. What is a branch in version control?

A. A physical tree branch in nature and the outdoors
B. A separate line of development diverging from the main codebase
C. A specific type of reported software bug and defect
D. A deployment stage in the release pipeline process

Answer: B

Q138. What is merging in version control?

A. Deleting branches in the system context
B. Creating new files during implementation
C. Combining companies throughout the project
D. Integrating changes from one branch into another

Answer: D

Q139. What is a repository?

A. A specific software testing environment and setup only in practice typically
B. A central location where version-controlled files and their history are stored
C. A production deployment server location and address as a standard approach
D. A physical library building for books and resources for quality purposes

Answer: B

Q140. What is a change request?

A. A formal proposal to modify a software configuration item
B. A new feature request from end users and clients
C. A request to change current employment job position
D. A software defect bug report document filed only

Answer: A

Q141. What is risk in software project management?

A. An uncertain event that could positively or negatively affect the project
B. A task that has already been fully completed and done in a systematic way
C. A documented project functional requirement and spec for the project goals
D. An absolute certainty in the project plan document across all phases

Answer: A

Q142. What is risk identification?

A. The process of determining which risks might affect the project
B. Intentionally creating new project risks and issues
C. Completely eliminating all identified project risks
D. Deliberately ignoring all project risks completely

Answer: A

Q143. What is risk analysis?

A. A source code implementation coding technique only
B. Analyzing application source code logic and syntax
C. A comprehensive software testing method approach
D. Evaluating identified risks to determine their probability and impact

Answer: D

Q144. What is risk mitigation?

A. Accepting all identified risks without any action
B. Taking actions to reduce the probability or impact of a risk
C. Deliberately ignoring all identified project risks
D. Avoiding all projects and work activities entirely

Answer: B

Q145. What is a risk register?

A. A physical cash register in a retail store space according to best practices
B. A hardware register for storing CPU data and values by the development process
C. A document listing identified risks, their analysis, and planned responses
D. A specific type of application source code file only within the system boundary

Answer: C

Q146. What are the two components used to assess risk?

A. Code and tests over the
B. Scope and quality
C. Probability and impact
D. Time and money and its

Answer: C

Q147. What is a risk response strategy?

A. Deleting the risk register and all documents too
B. Running away from all identified project risks now
C. A planned approach to address an identified risk
D. Deliberately ignoring the identified risk entirely

Answer: C

Q148. What is risk avoidance?

A. Accepting the risk without any planned action taken
B. Changing the project plan to eliminate a risk or its impact
C. Deliberately ignoring the identified risk in full
D. Avoiding the project entirely from the very start

Answer: B

Q149. What is risk transfer?

A. Transferring application source code ownership
B. Moving data to a different storage location setup
C. Shifting the impact of a risk to a third party
D. Moving to a new office building location downtown

Answer: C

Q150. What is risk acceptance?

A. Accepting all reported bugs as intended features only
B. Accepting all proposed requirement changes submitted
C. Accepting that the project will certainly fail overall
D. Acknowledging a risk and deciding to take no preemptive action

Answer: D

Q151. What is software security?

A. Locking the physical computer server room and door across all phases
B. Using strong passwords for user accounts only ever for the project goals
C. Protecting software from unauthorized access, use, modification, and destruction
D. Installing antivirus software applications only ever in a systematic way

Answer: C

Q152. What is authentication?

A. Verifying the identity of a user or system
B. Encryption within the system boundary
C. Firewall configuration by the development process
D. Authorization according to best practices

Answer: A

Q153. What is authorization?

A. Encryption over the entire lifecycle as defined by standards
B. Authentication and its related activities
C. Determining what permissions an authenticated user has
D. Decryption for all project stakeholders within the given

Answer: C

Q154. What is encryption?

A. Authentication in the engineering discipline for effective project outcomes
B. Converting data into a coded form to prevent unauthorized access
C. Compression in all development efforts by the project team members
D. Decryption and related components to achieve project objectives

Answer: B

Q155. What is a vulnerability in software?

A. A software defect fix and patch release and update
B. A specific automated test case and scenario and script
C. An intended software feature and product capability
D. A weakness that can be exploited to compromise security

Answer: D

Q156. What is a security threat?

A. A scheduled software update release event and launch as part of the methodology
B. A formal source code peer review session and meeting during the software lifecycle
C. A potential cause of an unwanted incident that may result in harm to a system
D. A new feature request from stakeholders and clients in the development process

Answer: C

Q157. What is SQL injection?

A. A database installation and setup procedure and guide
B. An attack that inserts malicious SQL code into application queries
C. A scheduled database backup and restore job and task
D. A standard SQL database query feature and capability

Answer: B

Q158. What is cross-site scripting (XSS)?

A. An attack where malicious scripts are injected into web pages viewed by other users
B. A JavaScript library for DOM manipulation and events throughout the project
C. A CSS framework for styling web content and layouts for the development team
D. A source code implementation coding technique and tool within the project scope

Answer: A

Q159. What is a firewall?

A. A type of computer virus or malware threat and attack during implementation
B. A general-purpose programming language tool and system for quality purposes
C. A physical wall constructed of fire bricks and mortar in the system context
D. A network security device that monitors and controls incoming and outgoing traffic

Answer: D

Q160. What is HTTPS?

A. A general-purpose programming language type and tool in practice typically
B. A specific type of database storage system and service as a standard approach
C. A comprehensive software testing framework and tool set by the organization
D. HTTP Secure — a protocol for secure communication over a network using encryption

Answer: D

Q161. What is the primary purpose of software engineering practices?

A. To eliminate the need for any project documentation
B. To maximize the number of programming languages used
C. To focus exclusively on writing source code quickly
D. To apply systematic methods to software development

Answer: D

Q162. Which term describes software that is free from defects?

A. Reliable software
B. Recursive software
C. Redundant software
D. Reflective software

Answer: A

Q163. What is a software development life cycle (SDLC)?

A. A programming language used for building systems
B. A network protocol for transferring project data
C. A framework defining steps in software development
D. A hardware specification for running applications

Answer: C

Q164. Which of these is a key challenge in software engineering?

A. Increasing the number of operating systems used
B. Managing changing project requirements over time
C. Ensuring hardware components are always upgraded
D. Reducing the physical size of server equipment

Answer: B

Q165. What does the term 'stakeholder' mean in software projects?

A. A testing tool used for automated regression tests
B. A developer who writes the majority of source code
C. A server that hosts the application in production
D. Anyone with an interest in the software project outcome

Answer: D

Q166. What is meant by software maintainability?

A. Number of features included in the first release
B. Amount of memory the software consumes at runtime
C. Ease of modifying software after initial deployment
D. Speed at which software executes user operations

Answer: C

Q167. Which activity comes first in most software development processes?

A. Performance testing and tuning
B. Source code implementation phase
C. Requirements gathering and analysis
D. System deployment and release

Answer: C

Q168. What is the role of a software engineer?

A. To design, develop, and maintain software systems
B. To manage financial accounting for the company
C. To install electrical wiring in office buildings
D. To manufacture physical computer hardware units

Answer: A

Q169. What does 'scalability' mean for a software system?

A. Requirement to rewrite code for every new feature
B. Tendency to decrease in performance under any load
C. Ability to handle growing amounts of work efficiently
D. Capability to run on a single device without issues

Answer: C

Q170. Which of the following best describes a software prototype?

A. The final production release of the application
B. An early working model of the proposed system
C. A hardware device used during system testing
D. A legal contract between client and developers

Answer: B

Q171. What is the waterfall model in software development?

A. A random unstructured method without any planning steps
B. A circular iterative approach with no defined ending point
C. A sequential linear development approach with distinct phases
D. A parallel development technique using multiple teams only

Answer: C

Q172. Which phase typically follows design in the waterfall model?

A. Implementation and coding
B. System deployment step
C. Requirements gathering
D. Project initiation phase

Answer: A

Q173. What is an iterative development process?

A. Repeating development cycles to refine the software product
B. Skipping testing phases to accelerate delivery timelines
C. Completing all development in a single pass without changes
D. Eliminating design activities to focus on coding tasks

Answer: A

Q174. What does the V-model emphasize in software development?

A. Eliminating all forms of documentation from the process
B. Combining all phases into a single continuous activity
C. Removing verification activities to save project time
D. Corresponding testing phase for each development stage

Answer: D

Q175. Which model allows returning to previous phases if needed?

A. Incremental model
B. Pure waterfall model
C. Big bang model only
D. Code and fix model

Answer: A

Q176. What is the main advantage of using a process model?

A. It eliminates the need for skilled software developers
B. It guarantees the software will have zero defect count
C. It removes all project risks from the development work
D. It provides structure and predictability to development

Answer: D

Q177. In which model is software delivered in small functional pieces?

A. Ad hoc coding method
B. Big bang development way
C. Incremental delivery model
D. Pure waterfall approach

Answer: C

Q178. What is a spiral model primarily concerned with?

A. Eliminating documentation
B. Avoiding testing phases
C. Reducing team size needs
D. Risk analysis and management

Answer: D

Q179. What is the prototyping model used for?

A. Replacing all testing with user observation sessions
B. Building early versions to clarify system requirements
C. Eliminating the need for any design documentation
D. Skipping requirements and jumping to final deployment

Answer: B

Q180. Which model is best suited for well-understood requirements?

A. Spiral model technique
B. Prototyping model plan
C. Waterfall model approach
D. Agile methodology way

Answer: C

Q181. What is the core principle of agile software development?

A. Following a strict sequential development process
B. Completing comprehensive documentation before coding
C. Avoiding customer collaboration during development
D. Responding to change over following a rigid plan

Answer: D

Q182. What is a sprint in Scrum methodology?

A. A method for deploying applications to production servers
B. A tool for measuring source code execution performance
C. A document describing all system requirements upfront
D. A fixed time period for completing a set of work items

Answer: D

Q183. Who is the Product Owner in Scrum?

A. The developer who writes the most lines of source code
B. The tester who executes all automated regression test suites
C. The person who manages the server infrastructure setup
D. The person responsible for maximizing product value delivered

Answer: D

Q184. What is a user story in agile development?

A. A formal contract between the client and development team
B. A detailed technical specification for database schema design
C. A short description of a feature from the user perspective
D. A comprehensive test plan covering all system components

Answer: C

Q185. What does the Agile Manifesto value most?

A. Contract negotiation over customer collaboration efforts
B. Following a plan over responding to change requests
C. Individuals and interactions over processes and tools
D. Comprehensive documentation over working software delivery

Answer: C

Q186. What is a daily standup meeting in agile?

A. A monthly review of the project budget and resources
B. A brief daily meeting to share progress and obstacles
C. A quarterly assessment of organizational strategy goals
D. A weekly planning session for the entire project scope

Answer: B

Q187. What is the purpose of a product backlog?

A. To list all desired features and work items for the product
B. To track employee attendance and working hours precisely
C. To document the organizational chart of the company fully
D. To store source code versions in a repository management tool

Answer: A

Q188. How long is a typical Scrum sprint?

A. One to two days only
B. One to four weeks
C. Eight to ten months
D. Six to twelve months

Answer: B

Q189. What is the Scrum Master's primary role?

A. To approve all design choices
B. To manage project budget
C. To facilitate the Scrum process
D. To write most of the code

Answer: C

Q190. What happens during a sprint review?

A. All unfinished tasks are permanently removed from scope
B. The team demonstrates completed work to stakeholders
C. The development environment is reset to default state
D. The project budget is reduced by senior management

Answer: B

Q191. What is requirements engineering in software development?

A. The process of discovering and documenting system needs
B. A strategy for deploying applications to cloud servers
C. A technique for writing efficient source code algorithms
D. A method for testing software performance under load

Answer: A

Q192. What is a functional requirement?

A. A specification of what the system should do for users
B. A plan for marketing the software product to customers
C. A schedule for training end users on system operations
D. A description of the hardware needed to run the system

Answer: A

Q193. What is a non-functional requirement?

A. A project schedule milestone for the delivery deadline
B. A feature that the system must implement for the user
C. A programming language choice for the development team
D. A constraint on how the system performs its functions

Answer: D

Q194. Who are the primary sources of software requirements?

A. Hardware manufacturers and network equipment vendors
B. Operating system kernel developers and maintainers
C. Compiler designers and programming language creators
D. Stakeholders and end users of the system being built

Answer: D

Q195. What is a Software Requirements Specification (SRS)?

A. A network diagram showing system topology and connections
B. A database schema defining all tables and relationships
C. A source code file containing the main application logic
D. A document that describes all system requirements in detail

Answer: D

Q196. What does 'requirements elicitation' mean?

A. Testing the software against its documented requirements
B. Gathering requirements from stakeholders and other sources
C. Deploying the finished software to the production server
D. Deleting unnecessary requirements from the project scope

Answer: B

Q197. Which of these is an example of a non-functional requirement?

A. The system shall generate monthly financial reports
B. The system shall send email notifications to users
C. The system shall respond to queries within two seconds
D. The system shall allow users to create new accounts

Answer: C

Q198. What is the purpose of requirements validation?

A. To ensure requirements accurately reflect stakeholder needs
B. To train end users on how to operate the new system
C. To deploy the software to the staging environment first
D. To write source code that implements the system features

Answer: A

Q199. What is a use case in requirements engineering?

A. A report on the system's performance test results
B. A list of programming languages used in the project
C. A diagram of the database schema and relationships
D. A description of how users interact with the system

Answer: D

Q200. Why is requirements traceability important?

A. It links requirements to design, code, and test artifacts
B. It eliminates the need for user acceptance testing work
C. It reduces the amount of memory the application uses
D. It speeds up the compilation of the source code files

Answer: A

Q201. What is the primary goal of software project management?

A. Installing and configuring the production server hardware equipment
B. Writing all the source code for the project without any team members
C. Designing the user interface layout and choosing the color schemes
D. Delivering software on time, within budget, and meeting quality standards

Answer: D

Q202. What is a project milestone?

A. A hardware component required for the server infrastructure
B. A significant checkpoint marking completion of a project phase
C. A type of programming error found during code review sessions
D. A daily task assigned to individual software developers on team

Answer: B

Q203. What is the purpose of a Gantt chart in project management?

A. To document all the source code changes made by developers
B. To track the number of defects found during system testing
C. To visually represent the project schedule and task timelines
D. To calculate the total budget required for hardware purchases

Answer: C

Q204. What does WBS stand for in project management?

A. Workflow Business Strategy
B. Wireless Bandwidth System
C. Web Based Software tool
D. Work Breakdown Structure

Answer: D

Q205. Who is responsible for overall project success?

A. The quality assurance tester only
B. The external client exclusively
C. The project manager leading the team
D. Each individual developer equally

Answer: C

Q206. What is project scope in software development?

A. The defined boundaries of what the project will deliver
B. The programming language selected for implementation
C. The total number of developers assigned to the team
D. The number of servers allocated for production use

Answer: A

Q207. What is resource allocation in project management?

A. Assigning people, tools, and budget to project activities
B. Designing the database schema for the system storage
C. Deleting unnecessary files from the project repository
D. Running automated tests on the application source code

Answer: A

Q208. What is the critical path in a project schedule?

A. A shortcut method for skipping non-essential project activities
B. The easiest set of tasks that any team member can complete alone
C. A backup plan executed only when the main project plan fails
D. The longest sequence of dependent tasks determining project duration

Answer: D

Q209. What is a project stakeholder?

A. Only the software developers who write the application source code
B. Just the project manager who oversees the entire development team
C. Exclusively the client who pays for the software development work
D. Anyone who has an interest in or is affected by the project outcome

Answer: D

Q210. What does project monitoring involve?

A. Tracking progress and comparing it against the original plan
B. Writing source code for the application features and modules
C. Conducting final user acceptance testing before deployment
D. Designing the user interface screens and navigation flows

Answer: A

Q211. What is software design in the development process?

A. A technique for deploying software to production servers
B. A strategy for gathering requirements from stakeholders
C. A method for testing software under heavy load conditions
D. The process of defining system structure and components

Answer: D

Q212. What is cohesion in software design?

A. The degree to which elements within a module belong together
B. The total number of lines of code in a software application
C. The speed at which the application responds to user actions
D. The amount of dependencies between different software modules

Answer: A

Q213. What is coupling in software design?

A. The speed at which the application compiles from source
B. The degree of interdependence between software modules
C. The total number of classes in the application codebase
D. The amount of documentation written for each module

Answer: B

Q214. What is the goal of good software design?

A. Neither cohesion nor coupling matter in quality design work
B. High coupling between modules and low cohesion within them
C. High cohesion within modules and low coupling between them
D. Equal cohesion and coupling across all system components

Answer: C

Q215. What is an interface in software design?

A. A network protocol for transmitting data between servers
B. A contract defining how components interact with each other
C. A database table storing application configuration settings
D. A visual screen layout designed for end user interactions only

Answer: B

Q216. What is modularity in software design?

A. Writing all code in a single file without any organization
B. Maximizing dependencies between all parts of the system
C. Dividing software into separate independent functional units
D. Avoiding any separation of concerns in the code structure

Answer: C

Q217. What is abstraction in software design?

A. Avoiding any simplification of the system's internal logic
B. Making all internal data structures visible to every module
C. Including every implementation detail in the public interface
D. Hiding complex details and showing only essential features

Answer: D

Q218. What is the purpose of a design pattern?

A. To increase the complexity of the software architecture
B. To provide reusable solutions to common design problems
C. To replace the need for requirements analysis activities
D. To create unique solutions that cannot be reused anywhere

Answer: B

Q219. What is encapsulation in object-oriented design?

A. Exposing all internal data to every part of the program
B. Allowing unrestricted access to all object internal states
C. Bundling data and methods together and restricting access
D. Separating data from the methods that operate on the data

Answer: C

Q220. What is a class diagram used for in software design?

A. To display the project timeline and scheduling milestones
B. To show the structure of classes and their relationships
C. To illustrate network topology and server configurations
D. To represent database queries and their execution plans

Answer: B

Q221. What is software architecture?

A. A detailed line-by-line description of the source code logic
B. A specific programming language used for building applications
C. A testing framework for validating user interface interactions
D. The high-level structure of a software system and its components

Answer: D

Q222. What is a client-server architecture?

A. A design pattern for sorting data in ascending order quickly
B. A system where all processing occurs on the user device only
C. A system where clients request services from a central server
D. A testing methodology for validating database query results

Answer: C

Q223. What is a layered architecture?

A. An architecture using a single flat structure without layers
B. An architecture organizing software into hierarchical layers
C. A deployment strategy for distributing code across servers
D. A testing technique for validating multi-step user workflows

Answer: B

Q224. What is a monolithic architecture?

A. A single unified application where all components are interconnected
B. A database architecture using multiple separate data storage nodes
C. A network topology with redundant connections between all nodes
D. A distributed system with many independent microservice components

Answer: A

Q225. What is the purpose of an architectural pattern?

A. To define specific algorithms for data processing and sorting
B. To provide proven solutions for common structural design problems
C. To specify the exact programming language for implementation
D. To determine the number of developers needed for a project

Answer: B

Q226. What is a microservices architecture?

A. Small independent services that communicate over network protocols
B. A single large application built as one deployable unit together
C. A programming style using only one function for all logic needs
D. A database design using only a single table for all data items

Answer: A

Q227. What is a pipe-and-filter architecture?

A. Data flows through processing components connected in sequence
B. A user interface pattern for displaying forms and input fields
C. All components execute simultaneously without any data flow order
D. A database technique for filtering query results by conditions

Answer: A

Q228. What is a repository architecture?

A. Data is distributed randomly across components without any order
B. Each component maintains its own isolated copy of all system data
C. Components never share any data and operate fully independently
D. Components share data through a central data store or repository

Answer: D

Q229. What does 'architectural style' refer to?

A. A family of systems sharing structural and behavioral properties
B. A method for writing unit tests for individual code functions
C. A specific color scheme used for the application user interface
D. A technique for compressing files to reduce storage space used

Answer: A

Q230. What is the role of an architect in software development?

A. Writing all the unit tests for the application source code now
B. Making high-level design decisions about system structure choices
C. Managing the project budget and timeline schedules for delivery
D. Conducting user interviews to gather system requirements only

Answer: B

Q231. What is user interface (UI) design?

A. Managing version control repositories for source code
B. Writing backend database queries for data retrieval tasks
C. Designing visual elements that users interact with on screen
D. Configuring server infrastructure for application hosting

Answer: C

Q232. What is usability in UI design?

A. The total number of colors used in the application theme
B. The programming language used to implement the interface
C. How easy and efficient it is for users to use the interface
D. The amount of data stored in the application database

Answer: C

Q233. What is a wireframe in UI design?

A. A basic visual layout showing the structure of a page
B. A database diagram showing tables and their relationships
C. A network map displaying server connections and topology
D. A fully functional prototype with all features implemented

Answer: A

Q234. What does 'responsive design' mean?

A. Design that adapts to different screen sizes and devices
B. Design that responds quickly to user input and actions
C. Design that avoids any visual elements on the interface
D. Design that uses only one fixed layout for all screens

Answer: A

Q235. What is the purpose of user feedback in UI design?

A. To reduce the number of features available to end users
B. To understand user needs and improve the interface design
C. To increase the amount of code in the application files
D. To eliminate the need for any future design improvements

Answer: B

Q236. What is a call-to-action (CTA) in UI design?

A. A prompt encouraging users to take a specific action step
B. A backend process running on the application server side
C. A database constraint enforcing data integrity rules now
D. A technical error message displayed during system failure

Answer: A

Q237. What is navigation in user interface design?

A. The server-side logic that processes user form inputs
B. The encryption algorithm protecting sensitive user data
C. The means by which users move through an application
D. The database indexing strategy for faster data queries

Answer: C

Q238. What is visual hierarchy in UI design?

A. Hiding important elements behind multiple menu layers
B. Removing all visual distinctions between page elements
C. Arranging elements to guide user attention by importance
D. Placing all elements at the same size and visual weight

Answer: C

Q239. What is the goal of accessibility in UI design?

A. Making interfaces only usable by experienced tech users
B. Reducing the number of interactive elements on a page
C. Restricting access to the interface based on user roles
D. Making interfaces usable by people with diverse abilities

Answer: D

Q240. What is a tooltip in user interface design?

A. A small popup providing information when hovering over an element
B. A large modal window requiring user input before proceeding further
C. A background service processing data without any user interaction
D. A database record storing user preferences and account settings

Answer: A

Q241. What is source code in software development?

A. A network configuration file for setting up server connections
B. The physical hardware on which software applications run today
C. Human-readable instructions written in a programming language
D. A project management document describing the timeline and budget

Answer: C

Q242. What is a compiler used for in software development?

A. Translating source code into machine-executable binary code
B. Creating database schemas and defining table relationships
C. Designing the user interface layout for web applications only
D. Managing project team members and assigning work to them

Answer: A

Q243. What is code refactoring?

A. Removing all comments and documentation from the source code
B. Deploying the application to a new production server environment
C. Restructuring existing code without changing its external behavior
D. Adding new features and functionality to the software product

Answer: C

Q244. What is version control in software development?

A. A system for tracking and managing changes to source code files
B. A tool for managing hardware inventory in the server data room
C. A method for testing the application performance under load now
D. A technique for designing the user interface with color themes

Answer: A

Q245. What is the purpose of code comments?

A. To explain the intent and logic of code for future readers
B. To encrypt the source code to prevent unauthorized access now
C. To increase the execution speed of the compiled application
D. To add new features without modifying any existing source code

Answer: A

Q246. What is an Integrated Development Environment (IDE)?

A. A database management system for storing application data now
B. A software tool combining editor, compiler, and debugger features
C. A physical room where developers work together on a project
D. A hardware device used for testing software on mobile devices

Answer: B

Q247. What is a syntax error in programming?

A. A violation of the programming language grammar and structure rules
B. A performance issue that causes the application to run very slowly
C. A logical mistake that produces incorrect output from the program
D. A design flaw in the system architecture causing scalability issues

Answer: A

Q248. What is debugging in software development?

A. Finding and fixing defects and errors in the source code logic
B. Deploying the application to the production server environment
C. Writing new source code to add features to the application
D. Designing the database schema for the application data store

Answer: A

Q249. What is a code review?

A. Running automated performance tests on the production environment
B. Deleting all source code and starting the project from scratch now
C. Installing software updates on all developer workstation machines
D. Examining source code by peers to find defects and improve quality

Answer: D

Q250. What is the purpose of coding standards?

A. To prevent any developer from modifying code written by other people
B. To ensure consistent code style and quality across the development team
C. To eliminate the need for testing by ensuring code is always correct
D. To restrict developers from using any external libraries or frameworks

Answer: B

Q251. What is the main purpose of software testing?

A. To design the user interface layout and visual style elements
B. To deploy the application to the production server environment
C. To find defects and verify the software meets its requirements
D. To write the source code for the application being developed

Answer: C

Q252. What is a test case in software testing?

A. A set of conditions and expected results for verifying functionality
B. A hardware component required for running the test environment
C. A programming language used for writing application source code
D. A project management document listing all team member assignments

Answer: A

Q253. What is unit testing?

A. Testing network connectivity between all production server nodes
B. Testing the entire integrated system from end to end completely
C. Testing the user interface for visual design and layout accuracy
D. Testing individual components or functions in isolation from others

Answer: D

Q254. What is integration testing?

A. Testing the project schedule against the planned delivery dates
B. Testing how multiple components work together as a combined unit
C. Testing individual functions in complete isolation from all others
D. Testing the application user interface for accessibility standards

Answer: B

Q255. What is a bug or defect in software?

A. A new feature requested by the client during the development phase
B. A hardware failure in the production server causing system downtime
C. A scheduled maintenance window for updating the operating system
D. A flaw causing the software to produce incorrect or unexpected results

Answer: D

Q256. What is regression testing?

A. Testing new features for the first time before release to production
B. Removing features from the software to simplify the application now
C. Retesting software after changes to ensure existing features still work
D. Writing new source code to implement additional system requirements

Answer: C

Q257. What is system testing?

A. Testing a single function or method in isolation from all other code
B. Testing only the network configuration of the production servers
C. Testing only the database queries for correctness and performance
D. Testing the complete integrated system against specified requirements

Answer: D

Q258. What is the difference between manual and automated testing?

A. Manual testing is done by humans; automated testing uses scripts and tools
B. Automated testing cannot find any bugs that manual testing would miss
C. Manual and automated testing produce identical results in all scenarios
D. Manual testing is always faster than automated testing in every situation

Answer: A

Q259. What is acceptance testing?

A. Testing the network bandwidth between the client and server nodes
B. Testing to determine if the system satisfies the acceptance criteria
C. Testing individual database tables for data integrity constraints
D. Testing the internal code structure for compliance with style guides

Answer: B

Q260. What is a test plan?

A. A database query for retrieving user account information data
B. A document describing the testing scope, approach, and schedule
C. A network diagram showing the production server architecture
D. A source code file containing the main application entry point

Answer: B

Q261. What is software maintenance?

A. Modifying software after delivery to correct faults or improve performance
B. Writing the initial source code for a brand new software application
C. Designing the system architecture before any development work begins
D. Gathering user requirements during the early project planning phases

Answer: A

Q262. What is corrective maintenance?

A. Fixing bugs and defects discovered after software has been deployed
B. Adapting software to work in a new operating system environment
C. Improving performance without changing any existing functionality
D. Adding brand new features requested by the users after deployment

Answer: A

Q263. What is adaptive maintenance?

A. Adding new features that were not in the original requirements set
B. Fixing defects found during production use of the deployed system
C. Modifying software to work in a changed or new environment context
D. Restructuring code to improve readability without behavior change

Answer: C

Q264. What is perfective maintenance?

A. Fixing critical bugs that prevent the system from functioning at all
B. Removing the software from production and replacing it with a new one
C. Modifying the software to run on a completely different platform now
D. Enhancing software by adding new features or improving existing ones

Answer: D

Q265. What percentage of total software costs does maintenance typically represent?

A. About twenty percent of total lifecycle costs
B. Over sixty percent of total lifecycle costs
C. Less than five percent of total lifecycle costs
D. Exactly ten percent of total lifecycle costs

Answer: B

Q266. What is preventive maintenance?

A. Making changes to prevent future problems and improve maintainability
B. Adding features only when explicitly requested by the project sponsor
C. Fixing bugs only after users report them in the production environment
D. Removing all documentation from the codebase to simplify the system

Answer: A

Q267. What is a software patch?

A. A full version upgrade adding many new features and capabilities now
B. A small update released to fix specific bugs or security issues quickly
C. A complete rewrite of the entire application from scratch using new code
D. A hardware replacement required to run updated software on the server

Answer: B

Q268. Why is documentation important for software maintenance?

A. It increases the file size of the project without adding any value
B. It is only useful during initial development and never after that
C. It helps maintainers understand the system to make changes correctly
D. It slows down maintenance by requiring reading before any changes

Answer: C

Q269. What is software evolution?

A. The continuous process of changing software throughout its lifetime
B. A one-time event that happens only during initial development phase
C. The process of permanently discontinuing a software product fully
D. A testing technique applied only before the first release of code

Answer: A

Q270. What is legacy software?

A. Brand new software developed using the latest programming frameworks
B. A type of testing framework used exclusively for mobile applications
C. Older software that remains in use despite outdated technology choices
D. A project management methodology for small startup development teams

Answer: C

Q271. What is quality assurance (QA) in software engineering?

A. A programming language designed for writing high-quality source code automatically
B. A specific testing technique used only during the final phase of development work
C. A hardware specification defining minimum requirements for running applications
D. Planned activities ensuring the development process produces quality software products

Answer: D

Q272. What is the difference between quality assurance and quality control?

A. QA focuses on processes; quality control focuses on product inspection and testing
B. QA and quality control are identical concepts with no meaningful differences at all
C. QA is done after release; quality control is done before development starts now
D. QA tests the product; quality control improves the development process activities

Answer: A

Q273. What is a software quality standard?

A. A hardware requirement for running the quality assurance testing tools now
B. A documented set of criteria that software must meet for acceptable quality
C. A project management template for scheduling development team meetings
D. A specific programming language recommended for building quality software

Answer: B

Q274. What is the purpose of a quality audit?

A. To deploy the application to the production environment for user access
B. To evaluate whether quality processes are being followed correctly by teams
C. To design the database schema and define table relationships for storage
D. To write source code for the application features and user interface now

Answer: B

Q275. What does ISO 9001 relate to in software engineering?

A. A quality management system standard applicable to software development
B. A specific programming language standard for embedded system software
C. A hardware interface specification for connecting peripheral devices now
D. A network protocol standard for transmitting data between computers

Answer: A

Q276. What is a defect in the context of quality assurance?

A. A positive user review about the software application performance
B. Any deviation from the expected behavior defined in the requirements
C. A new feature added to the software after the initial release date
D. A planned system downtime for performing scheduled maintenance work

Answer: B

Q277. What is the role of a QA engineer?

A. Managing the project budget and negotiating contracts with vendors now
B. Ensuring software quality through testing, reviews, and process improvement
C. Designing the network infrastructure for the production environment
D. Writing all the source code for the application without any supervision

Answer: B

Q278. What is peer review in quality assurance?

A. A management review of project budget and timeline progress reports
B. A single developer reviewing their own code without any external feedback
C. Team members examining each other's work to find defects and improvements
D. An automated tool scanning source code for syntax errors and warnings

Answer: C

Q279. What is a quality metric?

A. A hardware sensor measuring server temperature in the data center
B. A type of source code comment used for documenting class interfaces
C. A network monitoring tool tracking bandwidth usage and latency now
D. A measurable indicator of software quality attributes and characteristics

Answer: D

Q280. Why is quality important in software engineering?

A. It increases the number of defects found during production operations
B. It makes the source code longer and more difficult to understand now
C. It ensures user satisfaction, reduces costs, and builds customer trust
D. It slows down development without providing any measurable benefits

Answer: C

Q281. What are software metrics?

A. Quantitative measures used to assess software quality and processes
B. A type of programming language for building web applications today
C. A hardware specification for measuring processor clock speed rates
D. A project management tool for scheduling team meetings and events

Answer: A

Q282. What does Lines of Code (LOC) measure?

A. The amount of memory the software uses during runtime execution
B. The speed at which the application executes user requests today
C. The number of team members working on the software project now
D. The size of a software program by counting its source code lines

Answer: D

Q283. What is cyclomatic complexity?

A. A method for designing user interfaces for mobile application screens
B. A strategy for managing project budgets and resource allocations today
C. A technique for deploying software to cloud computing environments now
D. A measure of the number of independent paths through a program's code

Answer: D

Q284. Why are software metrics important?

A. They replace the need for experienced developers on the team now
B. They eliminate the need for any testing or quality assurance work
C. They provide objective data for making informed project decisions
D. They guarantee that all software products will be defect-free always

Answer: C

Q285. What is a product metric?

A. A measure of some attribute of the software product being developed
B. A financial metric tracking company revenue from software licenses
C. A hardware metric measuring server uptime and availability rates
D. A measure of the development process performance and efficiency now

Answer: A

Q286. What is a process metric?

A. A type of user interface element used in dashboard design layouts
B. A measure of the final product features and characteristics directly
C. A network measurement tool for tracking data transfer speed rates
D. A measure of the software development process effectiveness and quality

Answer: D

Q287. What does defect density measure?

A. The physical density of storage media used by the application today
B. The number of defects per unit size of the software being measured
C. The visual density of elements on the user interface screen layout
D. The network traffic density between the client and server systems

Answer: B

Q288. What is the purpose of measuring code coverage?

A. To determine what percentage of code is exercised by test suites
B. To count the total number of source code files in the project now
C. To track the number of developers contributing to the codebase
D. To measure the physical disk space occupied by the application

Answer: A

Q289. What is mean time between failures (MTBF)?

A. The time it takes to write one line of source code on average
B. The average time a system operates between consecutive failures
C. The average salary of developers working on the software project
D. The number of features added to each software release on average

Answer: B

Q290. What is a baseline in software measurement?

A. A default user interface theme applied when the application starts
B. A specific line of code that forms the foundation of the application
C. A reference point for comparing future measurements and tracking change
D. A minimum hardware requirement for running the software application

Answer: C

Q291. What is software configuration management (SCM)?

A. Managing and tracking changes to software throughout its lifecycle
B. A hardware setup process for servers in the data center facility
C. A programming technique for writing configuration file parsers now
D. A testing method for verifying system configuration settings only

Answer: A

Q292. What is version control in configuration management?

A. A technique for designing user interfaces with multiple theme options
B. A strategy for managing team member schedules and work assignments
C. A method for testing different versions of the software simultaneously
D. A system that records changes to files over time for later retrieval

Answer: D

Q293. What is a software baseline in configuration management?

A. A preliminary draft of the source code before any reviews take place
B. An informal agreement between developers about coding style preferences
C. A formally reviewed and agreed-upon specification serving as a reference
D. A temporary test environment used during development and testing only

Answer: C

Q294. What is a configuration item (CI)?

A. An artifact that is placed under configuration management and tracked
B. A physical server component that needs regular hardware maintenance
C. A project management tool for scheduling team meetings and events
D. A financial metric tracking development costs against the project budget

Answer: A

Q295. What is the purpose of a change control board (CCB)?

A. To conduct performance testing on the application before release to users
B. To design the user interface screens and navigation flows for the system
C. To evaluate and approve or reject proposed changes to the project baseline
D. To write all the source code for the application without any team input

Answer: C

Q296. What is a code repository?

A. A customer support center handling user complaints and feature requests
B. A central location where source code is stored and version controlled
C. A training facility where developers learn new programming languages
D. A physical library containing printed copies of programming textbooks

Answer: B

Q297. What is branching in version control?

A. Deploying the application to multiple production servers simultaneously
B. Creating a separate line of development from the main codebase path
C. Merging all team members' work into a single file without any review
D. Deleting old versions of the source code to save disk storage space

Answer: B

Q298. What is merging in version control?

A. Combining changes from different branches into a single unified branch
B. Splitting a single file into multiple smaller files for better organization
C. Creating a new repository from scratch without any existing source code
D. Deleting all branches except the main branch to simplify the repository

Answer: A

Q299. What is a commit in version control?

A. A snapshot of changes saved to the repository at a specific point in time
B. A financial transaction related to purchasing software licenses for tools
C. A verbal agreement between team members about the project scope and goals
D. A hardware upgrade performed on the development workstation machines now

Answer: A

Q300. What is a release in software configuration management?

A. A meeting where the team discusses upcoming features and priorities now
B. A training session for new developers joining the project team currently
C. A testing phase where all automated test suites are executed and reviewed
D. A version of the software distributed to users for installation and use

Answer: D

Q301. What is risk in software project management?

A. A completed task that has already been delivered successfully now
B. An uncertain event that could negatively impact the project outcome
C. A team meeting scheduled to discuss upcoming project milestones
D. A guaranteed problem that will definitely occur during development

Answer: B

Q302. What is risk identification?

A. The process of eliminating all possible risks from the project scope
B. The process of finding and documenting potential project risks early
C. A technique for writing source code without any potential defects
D. A method for deploying software to production without any testing

Answer: B

Q303. What is risk mitigation?

A. Taking actions to reduce the probability or impact of identified risks
B. Ignoring identified risks and hoping they will not occur during work
C. Removing all project constraints to eliminate potential risk factors
D. Increasing the likelihood of risks to test the team's response ability

Answer: A

Q304. What is a risk register used for?

A. Tracking employee attendance and work hours for payroll processing
B. Storing source code backups on external hard drives for recovery
C. Managing customer support tickets and feature request submissions
D. Recording identified risks with their assessments and response plans

Answer: D

Q305. What is risk probability?

A. The likelihood that a specific risk event will actually occur today
B. The guaranteed certainty that a risk will happen in every project
C. The cost of fixing a problem after it occurs in production later
D. The time required to complete the project from start to finish

Answer: A

Q306. What is risk impact?

A. The total budget allocated for the software development project
B. The number of team members assigned to work on the project now
C. The severity of consequences if a risk event actually materializes
D. The programming language selected for implementing the features

Answer: C

Q307. What is risk avoidance?

A. Increasing the risk probability to learn from potential failure events
B. Changing the project plan to eliminate the risk or its impact entirely
C. Accepting the risk and taking no action to address it at all currently
D. Transferring the risk to another team without any formal agreement

Answer: B

Q308. What is risk acceptance?

A. Actively working to increase the probability of the risk event occurring
B. Denying that any risks exist in the project scope and schedule plan
C. Transferring all project risks to external contractors without oversight
D. Acknowledging a risk exists and deciding not to take action against it

Answer: D

Q309. What is a contingency plan?

A. A training program for teaching developers new programming skills
B. A budget report showing how much money has been spent so far now
C. A predefined plan to execute if a specific risk event actually occurs
D. A daily standup meeting where team members share progress updates

Answer: C

Q310. Who is responsible for managing risks in a software project?

A. The project manager with support from the entire project team
B. The hardware vendor who provides the production server equipment
C. The external client who commissioned the software development
D. Only the most junior developer on the project team exclusively

Answer: A

Q311. What is software security engineering?

A. A strategy for managing team schedules and project milestone dates
B. Building software that continues to function correctly under attack
C. A technique for optimizing database query execution and performance
D. A method for testing user interface responsiveness and visual design

Answer: B

Q312. What is authentication in software security?

A. Designing the user interface for accessibility compliance only
B. Verifying the identity of a user or system requesting access
C. Testing the application for performance under heavy load now
D. Encrypting all data stored in the application database tables

Answer: B

Q313. What is authorization in software security?

A. Encrypting data during transmission between client and the server
B. Verifying the identity of a user attempting to log into the system
C. Testing the application for compatibility across web browsers now
D. Determining what actions an authenticated user is permitted to do

Answer: D

Q314. What is encryption used for in software security?

A. Compressing files to reduce storage space on the server drives
B. Protecting data by converting it into an unreadable format safely
C. Speeding up database query execution for better performance now
D. Formatting user interface elements for better visual appearance

Answer: B

Q315. What is a firewall in security?

A. A programming language designed for building secure applications
B. A system that monitors and controls network traffic based on rules
C. A project management tool for tracking team progress and tasks
D. A testing framework for validating software against requirements

Answer: B

Q316. What is a software vulnerability?

A. A documentation gap that makes the codebase harder to read now
B. A hardware component that needs replacement in the server rack
C. A weakness in software that can be exploited by an attacker now
D. A planned feature that has not been implemented yet in the code

Answer: C

Q317. What is the principle of least privilege?

A. Giving all users full administrative access to the entire system now
B. Granting maximum privileges to reduce the number of access errors
C. Removing all access controls to simplify the system architecture
D. Giving users only the minimum access rights needed for their tasks

Answer: D

Q318. What is a security patch?

A. A software update that fixes known security vulnerabilities quickly
B. A project management document describing the sprint backlog items
C. A decorative design element added to the user interface layout now
D. A hardware component that improves server processing speed rates

Answer: A

Q319. What is malware?

A. A project management metric tracking team productivity and output
B. A testing tool for measuring application performance under load
C. Software designed to damage, disrupt, or gain unauthorized access
D. A feature enhancement requested by users during product feedback

Answer: C

Q320. What is a security audit?

A. A performance test measuring application response time and speed
B. A code review focused only on visual design and layout of pages
C. A financial review of the project budget expenditure and savings
D. A systematic evaluation of security measures and vulnerabilities now

Answer: D

Q321. What is the primary goal of software engineering?

A. To write code as fast as possible
B. To develop reliable software systematically
C. To eliminate all bugs
D. To use the latest technology

Answer: B

Q322. Which of the following is a key attribute of good software?

A. Complexity
B. Maintainability
C. Obscurity
D. Rigidity

Answer: B

Q323. What is the software crisis?

A. A shortage of programmers
B. The difficulty of writing correct, understandable software on time and within budget
C. A virus that destroys software
D. The rapid decline in hardware prices

Answer: B

Q324. What does the term 'software process' refer to?

A. The act of compiling code
B. A set of activities that leads to the production of a software product
C. The process of installing software
D. A method for deleting software

Answer: B

Q325. Which activity is NOT part of a generic software process?

A. Specification
B. Development
C. Validation
D. Hardware manufacturing

Answer: D

Q326. What is a software product?

A. Only the executable code
B. Code, documentation, and associated data
C. Only the user manual
D. Only the source code files

Answer: B

Q327. What is meant by software dependability?

A. Software that depends on hardware
B. Software that is reliable, secure, and safe
C. Software that requires constant updates
D. Software written by a single developer

Answer: B

Q328. Who are the stakeholders in a software project?

A. Only the programmers
B. Anyone affected by or involved in the project
C. Only the customers
D. Only the project manager

Answer: B

Q329. What does the acronym CASE stand for in software engineering?

A. Computer-Aided Software Engineering
B. Central Application System Environment
C. Code Analysis and Structured Editing
D. Comprehensive Automated System Evaluation

Answer: A

Q330. What is the difference between generic and customized software products?

A. There is no difference
B. Generic products are sold to any customer; customized products are built for a specific client
C. Generic products are always free
D. Customized products cannot be modified

Answer: B

Q331. What is a software process model?

A. A physical model of a computer
B. A simplified representation of a software development process
C. A type of programming language
D. A hardware design specification

Answer: B

Q332. In the Waterfall model, can you return to a previous phase easily?

A. Yes, it encourages going back
B. No, it follows a strict linear sequence
C. Only during testing
D. Only on weekends

Answer: B

Q333. What is a prototype in software development?

A. The final product
B. An early working model used to explore ideas and gather feedback
C. A type of bug report
D. A deployment tool

Answer: B

Q334. What is the main advantage of the iterative development model?

A. It never requires documentation
B. It allows for early delivery of partial systems and gradual improvement
C. It eliminates the need for testing
D. It requires no planning

Answer: B

Q335. Which phase of the Waterfall model involves defining what the system should do?

A. Implementation
B. Requirements analysis and definition
C. System design
D. Maintenance

Answer: B

Q336. What is incremental development?

A. Developing the entire system at once
B. Developing and delivering the system in small portions called increments
C. Developing only the user interface
D. Delaying all development until requirements are final

Answer: B

Q337. Which process model is best suited when requirements are well understood and unlikely to change?

A. Agile model
B. Waterfall model
C. Spiral model
D. Prototype model

Answer: B

Q338. What does the term 'process activity' mean in software engineering?

A. A recreational team event
B. A fundamental step in a software development process such as specification or testing
C. A marketing campaign
D. A hardware installation step

Answer: B

Q339. What distinguishes throwaway prototyping from evolutionary prototyping?

A. They are the same approach
B. Throwaway prototypes are discarded after gathering feedback; evolutionary prototypes are refined into the final product
C. Throwaway prototyping takes longer
D. Evolutionary prototyping is always discarded

Answer: B

Q340. What is the purpose of the design phase in a process model?

A. To write marketing materials
B. To define the system architecture and component structure
C. To recruit developers
D. To delete old code

Answer: B

Q341. How many values are stated in the Agile Manifesto?

A. 2
B. 4
C. 6
D. 10

Answer: B

Q342. What does the Agile Manifesto value over comprehensive documentation?

A. Detailed contracts
B. Working software
C. Extensive planning
D. Formal processes

Answer: B

Q343. What is a Sprint Review in Scrum?

A. A meeting to fire underperforming team members
B. A meeting at the end of a Sprint where the team demonstrates completed work to stakeholders
C. A daily coding session
D. A meeting to write documentation

Answer: B

Q344. What is a retrospective in Agile?

A. A look at competitor products
B. A meeting where the team reflects on the process and identifies improvements
C. A code compilation step
D. A customer presentation

Answer: B

Q345. Which Agile principle emphasizes face-to-face communication?

A. Documentation first
B. The most efficient method of conveying information is face-to-face conversation
C. Email is preferred
D. All communication should be written

Answer: B

Q346. What is the role of the Product Owner in Scrum?

A. Writing all the code
B. Representing the customer and managing the Product Backlog
C. Managing servers
D. Designing the user interface

Answer: B

Q347. What is the purpose of a Software Requirements Specification (SRS)?

A. To write code
B. To document what the software should do and its constraints
C. To deploy software
D. To test the final product

Answer: B

Q348. What is a stakeholder in requirements engineering?

A. Only the CEO
B. Any person or organization that has an interest in or is affected by the system
C. Only the end user
D. Only the developer

Answer: B

Q349. What is a feasibility study in requirements engineering?

A. A study on team morale
B. An assessment of whether a proposed system is technically, economically, and operationally viable
C. A review of competitor products
D. A performance test

Answer: B

Q350. What does 'requirements validation' ensure?

A. That the code compiles
B. That the documented requirements accurately reflect stakeholder needs
C. That the hardware is working
D. That the team is on schedule

Answer: B

Q351. What is an example of a non-functional requirement?

A. The system shall allow users to log in
B. The system shall respond to queries within 2 seconds
C. The system shall display a list of products
D. The system shall allow users to add items to a cart

Answer: B

Q352. What is a requirements interview?

A. A job interview for developers
B. A technique where analysts ask stakeholders questions to understand their needs
C. A code review session
D. A testing technique

Answer: B

Q353. Why is requirements documentation important?

A. It is not important
B. It provides a shared reference that reduces misunderstandings and supports verification
C. It replaces the need for testing
D. It is only needed for legal reasons

Answer: B

Q354. What is the main purpose of project planning?

A. To write code
B. To define the project scope, schedule, and resources needed
C. To test the software
D. To deploy the final product

Answer: B

Q355. What is a milestone in project management?

A. A unit of distance
B. A significant point or event in the project schedule marking the completion of a major deliverable
C. A type of programming error
D. A software tool

Answer: B

Q356. What does the critical path represent in project scheduling?

A. The easiest path through the project
B. The longest sequence of dependent tasks that determines the minimum project duration
C. The shortest task in the project
D. A backup plan

Answer: B

Q357. What is effort estimation in software projects?

A. Estimating the weight of hardware
B. Predicting the amount of work (person-hours or person-months) needed to complete the project
C. Counting the number of employees
D. Estimating electricity costs

Answer: B

Q358. What is a project schedule?

A. A list of holidays
B. A timeline showing when project activities will be performed and deliverables completed
C. A team roster
D. A list of programming languages

Answer: B

Q359. What is the purpose of a project status report?

A. To advertise the product
B. To communicate the current state of the project, including progress, issues, and risks to stakeholders
C. To write code documentation
D. To create test cases

Answer: B

Q360. What does the term 'scope' mean in project management?

A. The range of a telescope
B. The boundaries of the project defining what will and will not be delivered
C. The number of developers on the team
D. The programming language used

Answer: B

Q361. What is a PERT chart?

A. A bar chart showing resource usage
B. A network diagram showing task dependencies and estimated durations for project scheduling
C. A pie chart showing budget allocation
D. A flow chart for code logic

Answer: B

Q362. What is the role of a project manager in software development?

A. Writing all the code
B. Planning, organizing, and controlling project activities to meet objectives within constraints
C. Only attending meetings
D. Designing the database

Answer: B

Q363. What is information hiding in software design?

A. Encrypting all data
B. Restricting access to module internals so that other modules interact only through well-defined interfaces
C. Hiding code from developers
D. Deleting documentation

Answer: B

Q364. What is a class diagram in object-oriented design?

A. A classroom schedule
B. A UML diagram showing classes, their attributes, methods, and relationships
C. A network topology diagram
D. A flowchart of code execution

Answer: B

Q365. What is the purpose of architectural design?

A. To choose office furniture
B. To define the overall structure and organization of the software system
C. To write unit tests
D. To deploy the software

Answer: B

Q366. What is stepwise refinement?

A. Refining sugar for cooking
B. A top-down design strategy that progressively decomposes a high-level design into more detailed components
C. Optimizing database queries
D. A testing technique

Answer: B

Q367. What is an interface in object-oriented design?

A. A monitor screen
B. A contract that specifies a set of methods that a class must implement
C. A type of database
D. A network protocol

Answer: B

Q368. What does 'high cohesion' mean in software design?

A. Modules that do many unrelated things
B. A module whose elements are closely related and focused on a single well-defined purpose
C. High coupling between modules
D. A module with many dependencies

Answer: B

Q369. What is a sequence diagram used for?

A. Showing the order of DNA sequences
B. Showing the order of interactions between objects over time
C. Showing the project schedule
D. Showing the database schema

Answer: B

Q370. What is a peer-to-peer architecture?

A. An architecture with a central server
B. An architecture where each node acts as both client and server, sharing resources directly
C. A hierarchical architecture
D. A single-user system

Answer: B

Q371. What is the role of middleware in software architecture?

A. To design user interfaces
B. To provide services that enable communication and data exchange between different applications or components
C. To write documentation
D. To test software

Answer: B

Q372. What is the pipe-and-filter architectural style?

A. A plumbing design pattern
B. An architecture where data flows through a series of processing components (filters) connected by channels (pipes)
C. A testing methodology
D. A security technique

Answer: B

Q373. What is scalability in software architecture?

A. The size of the codebase
B. The ability of a system to handle increased workload by adding resources without fundamental changes
C. The number of files in a project
D. The font size in the UI

Answer: B

Q374. What is a repository architectural style?

A. A version control system
B. An architecture where components share data through a central data store
C. A type of database
D. A code hosting platform

Answer: B

Q375. What is the purpose of an architectural view?

A. A scenic view from the office
B. A representation of the system architecture from a particular perspective or concern
C. A code review
D. A test report

Answer: B

Q376. What is a component in software architecture?

A. A hardware chip
B. A modular, deployable, and replaceable unit of software that encapsulates a set of related functions
C. A test case
D. A project document

Answer: B

Q377. What is the purpose of user-centered design?

A. Designing for developers only
B. Placing the needs, abilities, and limitations of end users at the center of the design process
C. Designing only the database
D. Focusing exclusively on aesthetics

Answer: B

Q378. What is the purpose of visual hierarchy in UI design?

A. Making all elements the same size
B. Organizing and prioritizing elements so users naturally see the most important information first
C. Hiding important information
D. Making the design more complex

Answer: B

Q379. What is a tooltip in UI design?

A. A physical tool tip
B. A small informational pop-up that appears when a user hovers over or focuses on an element
C. A debugging message
D. A server error message

Answer: B

Q380. What is white space (negative space) in UI design?

A. A bug in the interface
B. The empty areas between UI elements that improve readability and reduce visual clutter
C. Unused storage space
D. Deleted content areas

Answer: B

Q381. What is a variable naming convention?

A. A meeting about variable names
B. A set of rules for how variables should be named in code to improve readability
C. A type of data structure
D. A debugging technique

Answer: B

Q382. What is an IDE (Integrated Development Environment)?

A. A type of programming language
B. A software application that provides comprehensive tools for coding, debugging, testing, and building software in one interface
C. A database management system
D. A project management tool

Answer: B

Q383. What is the difference between a compiler and an interpreter?

A. They are the same thing
B. A compiler translates the entire source code at once into machine code; an interpreter translates and executes code line by line
C. Compilers are slower
D. Interpreters produce machine code files

Answer: B

Q384. What is version control used for in implementation?

A. Controlling the version of the operating system
B. Tracking and managing changes to source code over time, allowing collaboration and rollback
C. Controlling access to the internet
D. Managing database versions

Answer: B

Q385. What is a build process?

A. Constructing a building
B. The automated process of compiling source code, linking dependencies, and producing executable software
C. A meeting to plan features
D. A manual testing procedure

Answer: B

Q386. What is a test suite?

A. A hotel room for testers
B. A collection of test cases organized together to test a particular feature or aspect of the system
C. A single test case
D. A testing tool

Answer: B

Q387. What is a test oracle?

A. A fortune-telling tool
B. A mechanism or reference used to determine whether a test has passed or failed by comparing actual results to expected results
C. A database management tool
D. A type of programming language

Answer: B

Q388. What is the difference between a bug and a defect?

A. They are completely different concepts
B. They are generally used interchangeably to refer to a flaw in the software that causes incorrect behavior
C. Bugs are in hardware; defects are in software
D. Defects are intentional; bugs are not

Answer: B

Q389. What is preventive maintenance in software?

A. Preventing users from using the software
B. Making changes to software to prevent potential future problems before they occur
C. Preventing new features from being added
D. Uninstalling the software

Answer: B

Q390. What percentage of a software system's total lifecycle cost is typically spent on maintenance?

A. Less than 10%
B. 60% to 80% or more
C. Exactly 50%
D. Less than 5%

Answer: B

Q391. What is a patch in software maintenance?

A. A fabric patch for clothing
B. A small update released to fix specific bugs or security vulnerabilities without a full new version
C. A complete system rewrite
D. A new feature release

Answer: B

Q392. What triggers corrective maintenance?

A. A request for new features
B. The discovery of bugs or defects that need to be fixed
C. A change in management
D. A system upgrade

Answer: B

Q393. What triggers adaptive maintenance?

A. User complaints about the color scheme
B. Changes in the operating environment such as new hardware, operating systems, or regulations
C. Developer preferences
D. Marketing campaigns

Answer: B

Q394. What is a legacy system?

A. A system that was recently built
B. An older software system that remains in use because it fulfills a critical business need despite using outdated technology
C. A system with no users
D. A brand-new prototype

Answer: B

Q395. What is the purpose of a maintenance request?

A. Requesting a day off
B. A formal document describing a needed change, fix, or enhancement to an existing software system
C. Requesting new hardware
D. Requesting a team meeting

Answer: B

Q396. What is the purpose of quality assurance in software?

A. To find all bugs
B. To establish processes and standards that prevent defects and ensure quality throughout development
C. To replace testing
D. To increase project costs

Answer: B

Q397. What is a quality standard?

A. A high-quality flag
B. A documented set of criteria, guidelines, and best practices that products or processes must meet
C. A type of programming language
D. A testing tool

Answer: B

Q398. What is a software audit?

A. A financial audit of the company
B. An independent examination of software products or processes to assess compliance with standards and specifications
C. A code compilation step
D. A deployment procedure

Answer: B

Q399. What is a checklist in QA?

A. A grocery list
B. A predefined list of items or criteria to be verified during quality reviews to ensure completeness
C. A list of team members
D. A deployment schedule

Answer: B

Q400. What is process improvement in QA?

A. Buying better hardware
B. Systematically analyzing and enhancing development processes to increase product quality and efficiency
C. Hiring more developers
D. Increasing project budgets

Answer: B

Q401. What is a defect report?

A. A report card for developers
B. A document that describes a found defect, including steps to reproduce, expected behavior, and actual behavior
C. A financial report
D. A project plan

Answer: B

Q402. What is peer review in the context of QA?

A. A review of peer-to-peer networks
B. A process where colleagues examine each other's work products to identify defects and share knowledge
C. A performance review by managers
D. A customer review

Answer: B

Q403. What is the purpose of software metrics?

A. To count developers
B. To quantitatively measure attributes of software products and processes for decision-making and improvement
C. To measure hardware performance
D. To count meetings

Answer: B

Q404. What is the difference between size metrics and complexity metrics?

A. They are the same
B. Size metrics measure the quantity of code (like LOC); complexity metrics measure how difficult the code is to understand and maintain
C. Size is always more important
D. Complexity cannot be measured

Answer: B

Q405. What is defect removal efficiency (DRE)?

A. The speed of removing code
B. The percentage of defects found before software release compared to the total defects found before and after release
C. The efficiency of the development team
D. The speed of code compilation

Answer: B

Q406. What is the purpose of measuring lines of code (LOC)?

A. To determine developer pay
B. To estimate the size of a software system, which can be used for effort estimation and productivity measurement
C. To count the number of files
D. To measure code quality directly

Answer: B

Q407. What is a project metric?

A. A metric about the project manager
B. A measurement that tracks project progress and performance, such as schedule variance, cost variance, or team velocity
C. A metric about the building
D. A metric about company revenue

Answer: B

Q408. Why can LOC be misleading as a productivity metric?

A. LOC is always accurate
B. It penalizes efficient code that achieves more with fewer lines and varies greatly by programming language
C. LOC is never misleading
D. All lines of code are equally valuable

Answer: B

Q409. What is a baseline in software metrics?

A. A fishing line
B. A set of reference measurements collected from past projects used for comparison and estimation in future projects
C. A line of code
D. A base class

Answer: B

Q410. What is a direct metric versus an indirect metric?

A. They are the same
B. A direct metric is measured directly (like LOC); an indirect metric is derived from other measurements (like defect density)
C. Direct metrics are more complex
D. Indirect metrics are always inaccurate

Answer: B

Q411. What is the purpose of a version control system?

A. To control the version of the operating system
B. To track and manage changes to files over time, enabling collaboration and history tracking
C. To control file access permissions
D. To manage database versions

Answer: B

Q412. What is the purpose of a release in configuration management?

A. Releasing employees
B. A formally distributed version of the software that has been tested and approved for deployment to users
C. Releasing disk space
D. Releasing a news article

Answer: B

Q413. What is a repository in version control?

A. A physical storage room
B. A central storage location that contains all files, their version history, and metadata managed by the version control system
C. A library building
D. A data center

Answer: B

Q414. What is a change request in configuration management?

A. A request to change office furniture
B. A formal proposal to modify a configuration item, documenting the reason, impact, and details of the requested change
C. A request for a new computer
D. A request for vacation

Answer: B

Q415. What is a merge in version control?

A. Merging two companies
B. Combining changes from different branches or versions into a single unified version
C. Merging database tables
D. Merging network connections

Answer: B

Q416. What is the difference between risk avoidance and risk acceptance?

A. They are the same strategy
B. Risk avoidance changes plans to eliminate the risk; risk acceptance acknowledges the risk and chooses not to take action unless it occurs
C. Risk avoidance is always possible
D. Risk acceptance means ignoring risks

Answer: B

Q417. What is the difference between a risk and an issue?

A. They are the same thing
B. A risk is a potential future problem; an issue is a problem that has already occurred
C. Risks are always worse than issues
D. Issues cannot be managed

Answer: B

Q418. What is the probability of a risk?

A. Always 100%
B. The likelihood that the risk event will actually occur, typically expressed as a percentage or qualitative rating
C. Always 0%
D. The cost of the risk

Answer: B

Q419. Why is early risk identification important?

A. It is not important
B. Identifying risks early allows more time and options for developing effective responses before they become costly problems
C. Early identification wastes time
D. Risks should only be identified at the end

Answer: B

Q420. What is encryption in software security?

A. Compressing files
B. The process of converting data into a coded format that can only be read by someone with the proper decryption key
C. Deleting files permanently
D. Backing up data

Answer: B

Q421. What is a firewall in network security?

A. A wall that is on fire
B. A security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules
C. A physical wall in a building
D. A type of antivirus software

Answer: B

Q422. What is a vulnerability in software security?

A. A strong point in the software
B. A weakness or flaw in the software that could be exploited by an attacker to compromise the system
C. A feature request
D. A type of software update

Answer: B

Q423. What is multi-factor authentication (MFA)?

A. Using the same password multiple times
B. A security mechanism requiring users to provide two or more different types of verification to prove their identity
C. Multiple usernames
D. Multiple login pages

Answer: B

Q424. What is the purpose of access control in software?

A. Controlling physical building access
B. Restricting who can access what resources and what actions they can perform within the system
C. Controlling internet speed
D. Controlling screen brightness

Answer: B

Q425. What is phishing?

A. Going fishing
B. A social engineering attack that tricks users into revealing sensitive information by impersonating a trustworthy entity
C. A type of programming
D. A network protocol

Answer: B

Q426. What is data integrity in security?

A. The amount of data stored
B. Ensuring that data has not been altered, tampered with, or corrupted in an unauthorized manner
C. The speed of data transfer
D. The format of data storage

Answer: B

Q427. What is the purpose of security logging?

A. Logging into secure systems
B. Recording security-relevant events and actions to enable monitoring, auditing, and forensic analysis
C. Logging trees for wood
D. Creating log files for debugging

Answer: B

Q428. What is a Sprint Backlog in Scrum?

A. A list of all features ever requested for the product
B. The set of product backlog items selected for the current sprint plus a plan for delivering them
C. A document that tracks bugs found during testing
D. A report generated after each sprint is completed

Answer: B

Q429. What does the term 'iteration' mean in Agile development?

A. A single line of code written by a developer
B. A fixed-length time period during which a team works on a set of features and produces a working increment
C. The process of hiring new team members
D. A method for deploying software to production servers

Answer: B

Q430. What is the purpose of a burndown chart in Agile?

A. To track the budget spent on a project
B. To show the remaining work in a sprint or project over time
C. To list all the bugs found during development
D. To assign tasks to individual team members

Answer: B

Q431. What is the recommended size of a Scrum team?

A. Exactly 3 members
B. Between 20 and 50 members
C. Typically 5 to 9 members plus Scrum Master and Product Owner
D. There is no recommended size for Scrum teams

Answer: C

Q432. What is the difference between functional and non-functional requirements?

A. Functional requirements describe what the system should do while non-functional requirements describe how well it should perform
B. Functional requirements are optional while non-functional requirements are mandatory
C. Non-functional requirements describe features while functional requirements describe constraints
D. There is no difference; they are interchangeable terms

Answer: A

Q433. What is a requirements review meeting?

A. A meeting where developers write code based on requirements
B. A meeting where stakeholders and the team examine requirements documents to identify errors, ambiguities, and omissions
C. A meeting held only after the software is deployed to users
D. A meeting to decide the project budget and timeline

Answer: B

Q434. What is a work package in project management?

A. The entire project plan document
B. The smallest unit of work in a Work Breakdown Structure that can be estimated, scheduled, and monitored
C. A software package used for managing projects
D. The final deliverable shipped to the customer

Answer: B

Q435. What is the principle of separation of concerns in software design?

A. Keeping the development team separated into isolated groups
B. Dividing a program into distinct sections where each section addresses a separate concern or functionality
C. Ensuring that all code is written in a single file for simplicity
D. Separating the testing phase from the development phase

Answer: B

Q436. What is an activity diagram in software design?

A. A diagram showing the physical hardware components of a system
B. A flowchart-like diagram that shows the flow of activities and actions in a process or system
C. A chart showing project team member availability
D. A diagram that displays database table relationships

Answer: B

Q437. What does 'low coupling' mean in software design?

A. Modules have many dependencies on each other and share data extensively
B. Modules are relatively independent with minimal dependencies between them
C. The software has very few modules in total
D. All modules are combined into a single large component

Answer: B

Q438. What is a distributed system in software architecture?

A. A system where all code runs on a single computer
B. A system whose components are located on different networked computers that communicate and coordinate actions by passing messages
C. A system that is distributed to users via USB drives
D. A system that only works without an internet connection

Answer: B

Q439. What is a prototype in UI design?

A. The final production-ready version of the software
B. An early model or simulation of the user interface used to explore and test design ideas before development
C. A type of programming language used for building interfaces
D. A document listing all the bugs found in the interface

Answer: B

Q440. What is a modal dialog in UI design?

A. A full-screen window that replaces the entire application
B. A window or box that appears on top of the main content and requires user interaction before they can return to the main interface
C. A type of navigation menu that appears at the bottom of the screen
D. A background process that runs without any visual indication

Answer: B

Q441. What is a breadcrumb in UI design?

A. A type of animation effect applied to buttons
B. A navigation aid that shows the user's current location within the site hierarchy and allows navigation back to parent pages
C. A small cookie stored in the user's browser
D. A debugging tool for tracking user errors

Answer: B

Q442. What does 'affordance' mean in UI design?

A. The cost of developing the user interface
B. A design property that suggests how an element should be used, such as a button looking clickable
C. The number of colors used in the interface
D. The amount of text displayed on a single screen

Answer: B

Q443. What is user-centered design (UCD)?

A. A design approach where developers make all design decisions without user input
B. A design philosophy that places the end user's needs, preferences, and limitations at the center of each stage of the design process
C. A technique for designing database schemas based on user data
D. A project management method for scheduling design tasks

Answer: B

Q444. What is a runtime error in software development?

A. An error that occurs only during the design phase
B. An error that occurs while the program is executing, causing it to crash or produce incorrect results
C. An error in the project management timeline
D. An error that can only be detected by reading the source code manually

Answer: B

Q445. What is a logic error in programming?

A. An error that prevents the program from compiling
B. An error where the program runs without crashing but produces incorrect results due to a flaw in the algorithm or logic
C. An error caused by hardware malfunction
D. An error that only occurs when the program is deployed to production

Answer: B

Q446. What is a code repository used for?

A. Storing only compiled binary files for deployment
B. Storing, managing, and tracking changes to source code files in a centralized or distributed location
C. Providing a graphical interface for writing code
D. Running automated tests on the production server

Answer: B

Q447. What is pseudocode?

A. A real programming language used for production systems
B. An informal, high-level description of an algorithm using natural language and basic programming constructs
C. Encrypted source code that cannot be read by humans
D. A tool for automatically generating documentation from code

Answer: B

Q448. What is the purpose of indentation in source code?

A. It makes the program run faster by optimizing memory usage
B. It improves code readability by visually representing the structure and nesting of code blocks
C. It is required for all programming languages to compile successfully
D. It adds comments to the code automatically

Answer: B

Q449. What is a test environment?

A. The office where testers sit and work
B. A configured setup of hardware, software, and network used to execute tests that simulates the production environment
C. A programming language used exclusively for writing tests
D. The final version of the software deployed to end users

Answer: B

Q450. What is the purpose of smoke testing?

A. To test the software under extreme stress conditions
B. To perform a quick preliminary test to check whether the most critical functions of the software work before deeper testing begins
C. To test the software's behavior when the hardware overheats
D. To measure the performance of the software under normal load

Answer: B

Q451. What is alpha testing?

A. Testing performed by end users in their own environment
B. Testing performed by internal staff at the developer's site before release to external users
C. Testing performed automatically without any human involvement
D. The first test case written for any software project

Answer: B

Q452. What is beta testing?

A. Testing performed only by the development team using automated tools
B. Testing performed by a limited number of external users in their real environment before the final release
C. Testing that occurs after the software has been fully deployed and is in production
D. A type of unit testing focused on mathematical calculations

Answer: B

Q453. What is a test fixture in software testing?

A. A physical device used to test hardware components
B. A fixed state or set of preconditions used as a baseline for running tests to ensure consistent and repeatable results
C. A metric that measures the percentage of code tested
D. A report generated after all tests have been executed

Answer: B

Q454. What does 'test coverage' measure?

A. The number of testers assigned to a project
B. The degree to which the source code or requirements are exercised by a set of tests
C. The amount of time spent on testing activities
D. The cost of testing relative to the total project budget

Answer: B

Q455. What is software re-engineering?

A. Building completely new software from scratch without referencing existing systems
B. The process of examining and restructuring existing software to reconstitute it in a new form while preserving its functionality
C. A marketing strategy for promoting outdated software products
D. The process of removing all documentation from a legacy system

Answer: B

Q456. Why is regression testing important during software maintenance?

A. It only tests new features and ignores existing functionality
B. It ensures that changes or fixes to the software have not introduced new defects in previously working functionality
C. It measures the financial cost of maintaining the software
D. It is only performed when the entire software is being rewritten

Answer: B

Q457. What is the difference between software maintenance and software support?

A. They are identical terms with no difference
B. Maintenance involves modifying the software code and system while support involves helping users resolve issues and answering questions
C. Support involves changing the code while maintenance involves only documentation updates
D. Maintenance is performed by users while support is performed by developers

Answer: B

Q458. What is a code walkthrough in quality assurance?

A. The physical act of walking around the office while reviewing code
B. An informal review where a developer leads team members through code to find errors and share knowledge
C. An automated tool that scans code for security vulnerabilities
D. A formal audit conducted by external quality assessors

Answer: B

Q459. What is the difference between a defect and a failure in quality assurance?

A. They are exactly the same thing with no difference
B. A defect is a flaw in the software code or design, while a failure is the observable incorrect behavior when a defect is executed
C. A failure occurs during development while a defect occurs during testing
D. Defects are found by users while failures are found by developers

Answer: B

Q460. What is the purpose of a quality management plan?

A. To list all the features the software should have
B. To define the quality standards, processes, metrics, and responsibilities for ensuring that the project meets quality requirements
C. To schedule marketing activities for the software release
D. To calculate the total budget for the development project

Answer: B

Q461. What is defect density as a software metric?

A. The total number of developers on a project
B. The number of defects found per unit size of software, typically per thousand lines of code (KLOC)
C. The time it takes to fix a single defect
D. The percentage of the project budget spent on defect prevention

Answer: B

Q462. What is a pull request in version control?

A. A request to download the latest version of the software
B. A proposal to merge changes from one branch into another, allowing team members to review and discuss the changes before integration
C. A request to remove files from the repository
D. A notification that the server is pulling updates automatically

Answer: B

Q463. What is the purpose of a .gitignore file?

A. To list all the files that should be included in every commit
B. To specify patterns of files and directories that Git should not track or include in the repository
C. To store Git configuration settings for the repository
D. To record the history of all commits made to the repository

Answer: B

Q464. What is a merge conflict in version control?

A. When two repositories have the same name and cannot coexist
B. When two developers have made changes to the same part of a file on different branches and the system cannot automatically combine them
C. When a developer accidentally deletes the entire repository
D. When the version control server runs out of storage space

Answer: B

Q465. What is a software build in configuration management?

A. The physical construction of hardware to run the software
B. The process of compiling source code, linking dependencies, and packaging the result into an executable or deployable artifact
C. A meeting where the team discusses the next sprint
D. A document describing the software architecture

Answer: B

Q466. What is a contingency plan in risk management?

A. The original project plan that never changes
B. A predefined set of actions to be taken if a specific risk event occurs
C. A plan for hiring additional developers
D. A marketing strategy for the software product

Answer: B

Q467. What is risk transfer in software project management?

A. Moving the project to a different department
B. Shifting the responsibility or financial impact of a risk to a third party, such as through insurance or outsourcing
C. Ignoring the risk and hoping it does not occur
D. Transferring all project files to a new repository

Answer: B

Q468. What is the purpose of risk monitoring in a software project?

A. To eliminate all risks from the project permanently
B. To continuously track identified risks, watch for new risks, and evaluate the effectiveness of risk responses throughout the project
C. To create a final report after the project is complete
D. To assign blame when things go wrong

Answer: B

Q469. What is risk impact in software risk management?

A. The likelihood that a risk will occur during the project
B. The severity of the consequences if the risk event actually occurs
C. The number of team members affected by project risks
D. The total number of risks identified in the risk register

Answer: B

Q470. What is a risk assessment matrix?

A. A spreadsheet listing all team members and their salaries
B. A grid that maps risks by their probability of occurrence and severity of impact to prioritize which risks need attention
C. A mathematical formula for calculating project costs
D. A chart showing the project timeline and milestones

Answer: B

Q471. What is a security threat in software engineering?

A. A feature that improves the software's performance
B. Any potential danger that could exploit a vulnerability to cause harm to a system, organization, or users
C. A type of software license agreement
D. A coding standard that developers must follow

Answer: B

Q472. What is input sanitization in software security?

A. Physically cleaning the keyboard and mouse used for development
B. The process of cleaning and validating user input to remove or neutralize potentially harmful characters or code before processing
C. Deleting all user data from the database periodically
D. A network configuration technique for blocking external connections

Answer: B

Q473. What is the purpose of a requirements baseline in software engineering?

A. To set a maximum number of requirements allowed in a project
B. To establish a formally approved set of requirements that serves as the agreed-upon foundation for development
C. To create a backup copy of the source code repository
D. To determine the programming language for the project

Answer: B

Q474. What is a service in service-oriented architecture (SOA)?

A. A physical server that hosts websites
B. A self-contained unit of functionality that can be accessed remotely and used independently
C. A customer support team that helps users
D. A database table that stores user information

Answer: B

Q475. What is fault tolerance in software architecture?

A. The ability of a system to blame developers for errors
B. The ability of a system to continue operating properly even when some of its components fail
C. A technique for writing code without any bugs
D. The process of testing software until no faults remain

Answer: B

Q476. What is a dropdown menu in user interface design?

A. A menu that permanently displays all options on screen at all times
B. A UI element that reveals a list of options when activated, allowing the user to select one from the list
C. A type of error message that drops down from the top of the screen
D. A navigation bar fixed at the bottom of the page

Answer: B

Q477. What is a defect report (bug report) in software testing?

A. A marketing document describing software features
B. A formal document that records the details of a discovered defect including steps to reproduce, expected behavior, and actual behavior
C. A project schedule showing testing deadlines
D. A contract between the client and the development company

Answer: B

Q478. What is code churn as a software metric?

A. The total number of developers who have ever worked on the project
B. A measure of the amount of code that is added, modified, or deleted over a period of time
C. The speed at which the compiler processes source code files
D. The number of comments in the source code divided by total lines

Answer: B

Q479. What is a diff in version control?

A. A type of programming error that is difficult to find
B. A representation showing the differences between two versions of a file, highlighting added, removed, and changed lines
C. A command for deleting files from the repository permanently
D. A way to create a new branch in the repository

Answer: B

Q480. What is risk mitigation in software project management?

A. Completely eliminating all risks from the project
B. Taking proactive steps to reduce the probability or impact of an identified risk
C. Ignoring risks until they become actual problems
D. Transferring the entire project to another team

Answer: B

Medium Questions

480 questions

Q481. Which IEEE standard defines software engineering?

A. IEEE 754
B. IEEE 610.12
C. IEEE 802.11
D. IEEE 1394

Answer: B

Q482. What is the difference between software engineering and computer science?

A. CS primarily focuses on project management
B. They are fundamentally the same discipline
C. CS focuses on theory; SE focuses on practical development
D. SE primarily focuses on hardware architecture

Answer: C

Q483. Which of the following is a generic software process activity?

A. Software specification
B. Network setup
C. Hardware installation
D. Printing reports

Answer: A

Q484. What is 'software reuse'?

A. Using existing software components in new systems
B. Writing all source code from scratch every time
C. Permanently deleting outdated legacy software
D. Copying proprietary code without any permission

Answer: A

Q485. Which factor contributed most to the software crisis?

A. Overly simple user requirements specifications
B. Rapidly advancing hardware computing capabilities
C. An oversaturation of qualified trained programmers
D. Increasing software complexity outpacing development methods

Answer: D

Q486. What is a software engineering method?

A. A specific type of programming language tool
B. A classification of computing hardware types
C. A structured approach for developing software
D. A specialized device for testing code logic

Answer: C

Q487. What is the significance of the NATO Conference 1968?

A. The initial public launch of the internet era for the development team
B. The initial development and creation of the UNIX system
C. The original invention of the C programming language throughout the project
D. First formal discussion of the software crisis and software engineering

Answer: D

Q488. Which is NOT a type of software application?

A. Hardware software
B. System software
C. Web applications
D. Embedded software

Answer: A

Q489. What does CASE stand for in software engineering?

A. Computer-Aided Software Engineering
B. Common Architecture System Engineering
C. Coded Application Standard Environment
D. Central Application Software Environment

Answer: A

Q490. What is the role of a software engineer?

A. Writing source code exclusively and nothing else
B. Administering and managing databases exclusively
C. Designing, developing, testing, and maintaining software systems
D. Performing software testing activities exclusively

Answer: C

Q491. What are the four quadrants of the Spiral model?

A. Planning, Risk Analysis, Engineering, Evaluation
B. Gather, Analyze, Build, Ship within the project scope
C. Design, Code, Test, Review during the software lifecycle
D. Code, Test, Deploy, Maintain in the development process

Answer: A

Q492. What is RAD (Rapid Application Development)?

A. An incremental model emphasizing short development cycles of 60-90 days
B. A standardized software testing framework tool for the development team
C. A formal documentation standard for projects throughout the project
D. A general-purpose programming language format in the system context

Answer: A

Q493. Which model combines elements of the Waterfall and Iterative models?

A. Spiral model
B. Big Bang model
C. RAD
D. Incremental model

Answer: A

Q494. What is the incremental model?

A. A model that skips the planning phase for quality purposes
B. Building the complete system all at once during implementation
C. A model used only for testing purposes in practice typically
D. Delivering the system in increments, each adding functionality

Answer: D

Q495. In the V-Model, what corresponds to the coding phase?

A. Integration testing
B. System testing
C. Unit testing
D. Acceptance testing

Answer: C

Q496. What is evolutionary development?

A. Developing software that remains unchanged permanently as a standard approach
B. Only creating project plans without writing code by the organization
C. Developing an initial version, exposing it to users, and refining through versions
D. Focusing solely on hardware evolution processes at every stage

Answer: C

Q497. What is the main advantage of the incremental model?

A. The model requires absolutely no testing phase in a systematic way
B. Working software is produced early and feedback can be incorporated
C. No formal planning phase is ever required across all phases
D. It is always the fastest model available today for the project goals

Answer: B

Q498. What is a phase gate in process models?

A. An established coding standard for the team
B. A specific category of reported software bug
C. A review point at the end of each phase before proceeding
D. A physical entry gate for developer access

Answer: C

Q499. Which model is best suited for large, mission-critical projects?

A. Code and Fix
B. Big Bang accord
C. Spiral model
D. RAD within the

Answer: C

Q500. What is throwaway prototyping?

A. A category of finalized production product only and its related activities
B. Retaining and maintaining all prototypes built by the development process
C. A formal software testing methodology approach over the entire lifecycle
D. Building a prototype to understand requirements, then discarding it

Answer: D

Q501. What is the role of the Scrum Master?

A. Writing all of the source code for the team
B. Designing the relational database architecture
C. Managing the overall project budget allocations
D. Facilitating the Scrum process and removing impediments

Answer: D

Q502. What is Extreme Programming (XP)?

A. A video game development methodology only throughout the project
B. An Agile methodology emphasizing engineering practices like pair programming and TDD
C. A project management scheduling tool only in the system context
D. A unit testing automation framework only during implementation for quality purposes

Answer: B

Q503. What is a Product Backlog?

A. A finalized list of all completed work tasks
B. An ordered list of everything needed in the product
C. A comprehensive list of known bugs and issues
D. A physical hardware inventory listing sheet

Answer: B

Q504. What is the purpose of a Sprint Retrospective?

A. To reflect on the sprint and identify improvements
B. To plan all the tasks for the upcoming sprint
C. To demonstrate completed features to stakeholders
D. To write all required project documentation

Answer: A

Q505. What is Kanban in software development?

A. A visual workflow management method using boards and cards
B. A general-purpose programming language tool
C. A comprehensive testing automation framework
D. A relational database management system

Answer: A

Q506. What is 'velocity' in Agile?

A. The amount of work a team can complete in a sprint
B. The physical typing speed of the developers
C. The network communication transfer bandwidth
D. The code compilation and build execution speed

Answer: A

Q507. What is pair programming?

A. Two programmers working together at one workstation
B. Two people using the same computer for different tasks
C. Two separate teams coding on different features
D. Programming in pairs of related source files

Answer: A

Q508. What is a Sprint Review?

A. A comprehensive team evaluation process event
B. A meeting to demonstrate completed work to stakeholders
C. A formal source code review session for the team
D. An individual performance review meeting session

Answer: B

Q509. What is the 'Definition of Done' in Scrum?

A. A shared understanding of what it means for work to be complete
B. When the source code compiles without errors
C. When the project manager gives final approval
D. When all documentation has been written and reviewed

Answer: A

Q510. What is Continuous Integration in Agile?

A. Integrating code once at the very end of the project in practice typically
B. Frequently merging code changes into a shared repository with automated testing
C. Performing only manual source code reviews periodically as a standard approach
D. Conducting annual system integration events for release by the organization

Answer: B

Q511. What is the difference between verification and validation in requirements?

A. Verification checks if we're building the product right; validation checks if we're building the right product
B. They are fundamentally identical concepts entirely for the project goals according to best practices
C. Verification is testing code; validation is writing code within the system boundary
D. Validation always occurs before verification ever does by the development process and its related activities

Answer: A

Q512. What is requirements traceability?

A. Tracking individual developer working hours over the entire lifecycle
B. The ability to trace each requirement through design, implementation, and testing
C. An established coding standard for the project for all project stakeholders
D. A specific debugging and troubleshooting technique as defined by standards

Answer: B

Q513. What is a use case diagram?

A. A process execution flowchart diagram only within the given constraints
B. A relational database schema diagram layout in all development efforts
C. A UML diagram showing actors and their interactions with the system
D. A network topology and routing diagram map by the project team members

Answer: C

Q514. What is requirements prioritization?

A. Permanently deleting all unimportant requirements
B. Ranking requirements by importance, risk, or business value
C. Grouping requirements by assigned developer team
D. Alphabetical ordering of all requirements listed

Answer: B

Q515. What is prototyping used for in requirements engineering?

A. Delivering the final production product to users
B. Performing application security penetration tests
C. Conducting system performance load testing runs
D. Helping stakeholders visualize and validate requirements

Answer: D

Q516. What is a scenario in requirements engineering?

A. A real-life example describing how a system might be used
B. An established coding guideline for the teams
C. A comprehensive deployment plan document only
D. A specific automated test execution script only

Answer: A

Q517. What is requirements negotiation?

A. Rejecting all stakeholder proposed requirements
B. Resolving conflicts between stakeholders regarding requirements
C. Demanding that all requirements absolutely be met
D. Deliberately ignoring all stakeholder input

Answer: B

Q518. What is domain analysis in requirements engineering?

A. Studying specific programming language syntax in the engineering discipline
B. Studying the application domain to understand its concepts and constraints
C. Analyzing the database schema domain structure and related components
D. Analyzing network domain name configurations to achieve project objectives

Answer: B

Q519. What is the IEEE 830 standard?

A. A networking communication protocol standard
B. A standard for writing software requirements specifications
C. A software testing methodology and standard
D. A hardware design and manufacturing standard

Answer: B

Q520. What is a constraint in requirements engineering?

A. A defect discovered in the source code base for effective project outcomes
B. A restriction or limitation on the system or development process
C. A programming language built-in feature set in the development process
D. A specific type of software test case only as part of the methodology

Answer: B

Q521. What is the COCOMO model?

A. A comprehensive software testing methodology and its related activities
B. A creational object-oriented design pattern over the entire lifecycle
C. A popular chocolate confectionery brand name by the development process
D. A model for estimating software development effort based on lines of code

Answer: D

Q522. What is function point analysis?

A. A method for measuring software size based on functionality delivered to users
B. A specific software testing automation technique for all project stakeholders
C. A systematic debugging and tracing methodology within the given constraints
D. Counting total functions in application code as defined by standards

Answer: A

Q523. What is Earned Value Management (EVM)?

A. A corporate employee salary payment system in all development efforts
B. A technique for measuring project performance by comparing planned vs actual progress
C. An established coding standard and guideline set and related components
D. A specific software testing coverage metric only by the project team members

Answer: B

Q524. What is the difference between PERT and CPM?

A. PERT does not produce any chart output at all for effective project outcomes
B. The two techniques are completely identical overall to achieve project objectives
C. CPM is exclusively used for software projects in the engineering discipline
D. PERT uses probabilistic time estimates; CPM uses deterministic time estimates

Answer: D

Q525. What is slack (float) in project scheduling?

A. A specific category of software defect type during the software lifecycle
B. Additional unnecessary source code that was written in the development process
C. Being unproductive and lazy at work always as part of the methodology
D. The amount of time a task can be delayed without delaying the project

Answer: D

Q526. What is a project baseline?

A. The very first line of application source code for the development team
B. An approved version of the project plan used to measure performance
C. A performance testing benchmark result value throughout the project
D. The initial starting point for all coding work within the project scope

Answer: B

Q527. What is the purpose of a project charter?

A. To write application source code for features in the system context
B. To perform software testing and validation runs during implementation
C. To formally authorize the project and define its objectives and stakeholders
D. To deploy software to production server systems for quality purposes

Answer: C

Q528. What is the difference between effort and duration in project estimation?

A. Effort is total person-hours; duration is calendar time
B. The two terms are exactly the same thing always
C. Effort is always measured in monetary currency
D. Duration is always longer than effort estimates

Answer: A

Q529. What is a risk in project management?

A. An uncertain event that could positively or negatively affect the project
B. An absolute certainty in the project plan only in practice typically
C. A documented project functional requirement by the organization
D. A task that has already been fully completed as a standard approach

Answer: A

Q530. What is the purpose of a status report?

A. To design relational database architectures in a systematic way
B. To write application source code for features at every stage
C. To compile and build software executable files across all phases
D. To communicate project progress, issues, and plans to stakeholders

Answer: D

Q531. What is the Singleton design pattern?

A. A pattern that ensures a class has only one instance with a global access point
B. A comprehensive software testing design pattern by the development process
C. A relational database schema design pattern type and its related activities
D. A pattern designed for creating multiple instances within the system boundary

Answer: A

Q532. What is the Observer design pattern?

A. A pattern where objects are notified of state changes in another object
B. A creational object instantiation design pattern as defined by standards
C. A structural composition and wrapping pattern for all project stakeholders
D. A system security and access control pattern over the entire lifecycle

Answer: A

Q533. What is the Factory Method pattern?

A. A pattern that defines an interface for creating objects, letting subclasses decide which class to instantiate
B. A specialized debugging and tracing tool suite to achieve project objectives in the engineering discipline
C. A comprehensive software testing method and tool by the project team members and related components
D. A physical manufacturing production process flow within the given constraints in all development efforts

Answer: A

Q534. What is the MVC design pattern?

A. A comprehensive software testing pattern type as part of the methodology
B. A specific type of program variable declaration for effective project outcomes
C. Model-View-Controller, separating data, presentation, and control logic
D. A software deployment automation pattern type in the development process

Answer: C

Q535. What is the Strategy design pattern?

A. A specific software testing strategy and plan within the project scope
B. A project management planning approach method during the software lifecycle
C. A pattern that defines a family of algorithms and makes them interchangeable
D. A software deployment automation strategy type for the development team

Answer: C

Q536. What are the three categories of GoF design patterns?

A. Input, Process, Output in the system
B. Creational, Structural, Behavioral
C. Design, Code, Test during implementation
D. Easy, Medium, Hard throughout the project

Answer: B

Q537. What is the Open/Closed Principle?

A. Software entities should be open for extension but closed for modification
B. Never closing running desktop applications ever as a standard approach
C. Always keeping file handles open in the code in practice typically
D. Using only open source software for projects for quality purposes

Answer: A

Q538. What is the Liskov Substitution Principle?

A. A specific naming convention for all variables by the organization at every stage
B. A comprehensive software testing principle type across all phases in a systematic way
C. A relational database normalization principle for the project goals according to best practices
D. Objects of a superclass should be replaceable with objects of a subclass without affecting correctness

Answer: D

Q539. What is the DRY principle?

A. Keep code dry and clean within the system boundary
B. Delete Redundant Yields by the development process
C. Don't Repeat Yourself — avoid code duplication
D. Develop Robust Yields and its related activities

Answer: C

Q540. What is a class diagram?

A. A project timeline for all project stakeholders within the given constraints
B. A network diagram in all development efforts by the project team members
C. A classroom layout over the entire lifecycle as defined by standards
D. A UML diagram showing classes, attributes, methods, and relationships

Answer: D

Q541. What is microservices architecture?

A. An architecture where the application is composed of small, independent services
B. A traditional monolithic design approach to apps and related components
C. A single centralized database system approach to achieve project objectives
D. One single large monolithic service application by the project team members

Answer: A

Q542. What is the difference between SOA and microservices?

A. SOA is a much newer architectural approach style as part of the methodology
B. SOA uses enterprise service bus for communication; microservices use lightweight protocols
C. Microservices are generally larger in scope overall for effective project outcomes
D. The two approaches are completely identical overall in the engineering discipline

Answer: B

Q543. What is an architectural style?

A. A named collection of architectural design decisions applicable to recurring design problems
B. A user interface visual style and theme design during the software lifecycle
C. A specific coding style and convention guideline in the development process
D. A project documentation format and standard set within the project scope

Answer: A

Q544. What is the Model-View-ViewModel (MVVM) pattern?

A. An architectural pattern separating UI, presentation logic, and business logic/data
B. A relational database normalization pattern type for the development team
C. A comprehensive software testing pattern design throughout the project
D. A production deployment automation pattern type in the system context

Answer: A

Q545. What are architectural views?

A. Database views defined with SQL query language for quality purposes
B. User interface screens and layout page designs in practice typically
C. Different perspectives of a system architecture for different stakeholders
D. Screen views displayed to the end users only during implementation

Answer: C

Q546. What is the 4+1 View Model?

A. A comprehensive software testing model and plan by the organization at every stage
B. A relational database normalization model type across all phases in a systematic way
C. An architecture description using Logical, Process, Development, Physical views plus Scenarios
D. Five physical computer display monitors together as a standard approach

Answer: C

Q547. What is a message broker in architecture?

A. A general-purpose programming language and tool within the system boundary
B. A specific type of database storage system tool according to best practices
C. An intermediary program that translates messages between different systems
D. A person who physically delivers messages by hand for the project goals

Answer: C

Q548. What is the publish-subscribe pattern?

A. A messaging pattern where publishers send messages to subscribers through a topic
B. An established coding standard and guideline set over the entire lifecycle
C. A relational database design pattern type only and its related activities
D. A physical magazine subscription service plan by the development process

Answer: A

Q549. What is a service-oriented architecture (SOA)?

A. An architecture where services communicate over a network to provide functionality
B. A hardware system architecture blueprint only within the given constraints
C. A single isolated standalone service application as defined by standards
D. A traditional monolithic application design style for all project stakeholders

Answer: A

Q550. What is an API Gateway?

A. A physical security entry gate device system in all development efforts
B. A hardware network switching device and unit and related components
C. A server that acts as a single entry point for a set of microservices
D. A relational database gateway and adapter tool by the project team members

Answer: C

Q551. What are Nielsen's 10 usability heuristics?

A. Ten specific software testing methodology types
B. Ten established coding standards and rule sets
C. Guidelines for evaluating user interface design quality
D. Ten strict programming coding rules for teams

Answer: C

Q552. What is Fitts' Law?

A. An established coding law and standard rule only and its related activities over the entire lifecycle
B. A fundamental law of classical physics and motion within the system boundary by the development process
C. A relational database normalization law and rule as defined by standards for all project stakeholders
D. The time to reach a target is proportional to the distance and inversely proportional to the target size

Answer: D

Q553. What is a heuristic evaluation?

A. A mathematical algorithm analysis method only within the given constraints
B. A usability inspection method where experts evaluate UI against established principles
C. A formal source code peer review session only by the project team members
D. A system performance load testing method only in all development efforts

Answer: B

Q554. What is the difference between UI and UX?

A. UX is only about writing source code logic only as part of the methodology in the development process
B. The two concepts are fundamentally the same thing and related components to achieve project objectives
C. UI focuses on visual design; UX encompasses the entire user experience including emotions and satisfaction
D. UI is always more important than UX design work in the engineering discipline for effective project outcomes

Answer: C

Q555. What is A/B testing in UI design?

A. Testing against two different database systems
B. Testing on two different mobile device models
C. Comparing two versions of a design to see which performs better
D. Testing two different API endpoints separately

Answer: C

Q556. What is the concept of 'affordance' in UI design?

A. The financial cost of design work and effort
B. A particular application color scheme and theme
C. A specific type of user interface animation style
D. A visual clue that suggests how an element can be used

Answer: D

Q557. What is a design system?

A. A comprehensive software testing system and suite for the development team
B. A computer operating system installation setup during the software lifecycle
C. A collection of reusable components, guidelines, and standards for consistent design
D. A specific application coding framework tool type within the project scope

Answer: C

Q558. What is the principle of 'progressive disclosure'?

A. Showing every option and detail at once to users throughout the project
B. Hiding absolutely all system information entirely in the system context
C. Showing only essential information initially, revealing more as needed
D. Disclosing security information to all end users during implementation

Answer: C

Q559. What is a user persona?

A. A specific type of user profile avatar image in practice typically
B. A user login name and credentials identifier for quality purposes
C. A user uploaded profile picture image file as a standard approach
D. A fictional character representing a key user group based on research data

Answer: D

Q560. What is the Z-pattern layout?

A. A relational database query plan pattern and type across all phases in a systematic way
B. A comprehensive software testing pattern approach for the project goals
C. A layout following the natural eye scanning pattern from top-left to bottom-right in a Z shape
D. A zigzag code structure in the code base layout by the organization at every stage

Answer: C

Q561. What is continuous integration (CI)?

A. Manually integrating code without any automation for the development team
B. Occasional and infrequent code merging events during the software lifecycle
C. The practice of frequently merging code changes with automated building and testing
D. Annual scheduled system integration events only within the project scope

Answer: C

Q562. What is continuous deployment (CD)?

A. Annual scheduled deployment release events only in the system context
B. Deployment without any automated testing at all during implementation
C. Manual deployment to production server systems throughout the project
D. Automatically deploying every code change that passes automated tests to production

Answer: D

Q563. What is the difference between compilation and interpretation?

A. Compilation is always done only at runtime always at every stage across all phases in a systematic way
B. Compilation translates entire code to machine code before execution; interpretation translates line by line during execution
C. Interpretation is always faster than compilation is as a standard approach by the organization
D. The two processes are fundamentally the same thing for quality purposes in practice typically

Answer: B

Q564. What is pair programming?

A. Programming for exactly two hours at a time only
B. Programming on two separate display monitors only
C. Two developers working together at one workstation
D. Two separate teams coding independently on tasks

Answer: C

Q565. What is code smell?

A. An actual physical smell from hardware component
B. A simple code formatting and indentation error
C. A symptom in the source code that may indicate a deeper problem
D. A type of computer virus or malware threat only

Answer: C

Q566. What is static code analysis?

A. Executing and running the application code live
B. Analyzing code without executing it to find potential errors
C. Dynamic runtime testing of the application code
D. Performance load testing of the application code

Answer: B

Q567. What is Git branching?

A. A comprehensive software testing method type
B. Creating separate lines of development within a repository
C. A type of sorting algorithm technique approach
D. Physical branches of a tree in nature settings

Answer: B

Q568. What is code coverage?

A. A project documentation completeness metric type
B. Physical coverage of printed code documents only
C. A metric measuring the percentage of code executed during testing
D. A security protection and control measure type

Answer: C

Q569. What is the DRY principle in coding?

A. Delete Redundant Yields according to best practices
B. Develop Robust Yields within the system boundary
C. Keep code moisture-free for the project goals
D. Don't Repeat Yourself — avoid duplicating code or logic

Answer: D

Q570. What is a linter?

A. A tool that analyzes code for potential errors, style violations, and best practices
B. A type of clothing fabric material only exactly by the development process
C. A source code compilation tool program and build over the entire lifecycle
D. A specialized debugging and tracing tool for code and its related activities

Answer: A

Q571. What is boundary value analysis?

A. A testing technique focusing on values at the edges of input ranges
B. A software architecture design technique approach throughout the project
C. Analyzing overall project scope boundary limits for the development team
D. A source code implementation technique approach in the system context

Answer: A

Q572. What is equivalence partitioning?

A. A creational object-oriented design pattern type for quality purposes
B. Dividing input data into groups that should be treated the same by the software
C. A source code implementation technique approach in practice typically
D. Equal distribution of work across team members during implementation

Answer: B

Q573. What is the difference between verification and validation in testing?

A. Verification is entirely optional in all practice in a systematic way for the project goals
B. The two concepts are fundamentally the same thing as a standard approach by the organization
C. Verification checks if the product is built right; validation checks if the right product is built
D. Validation always occurs before verification process at every stage across all phases

Answer: C

Q574. What is system testing?

A. Testing just the operating system installation only within the system boundary
B. Testing only individual isolated code units first according to best practices
C. Testing the complete integrated system to verify it meets requirements
D. Testing physical hardware system components only by the development process

Answer: C

Q575. What is performance testing?

A. Testing how the system performs under various conditions of load and stress
B. Testing the overall user experience quality level over the entire lifecycle
C. Testing overall source code quality metrics only and its related activities
D. Testing application security vulnerabilities only as defined by standards

Answer: A

Q576. What is stress testing?

A. Testing comprehensive project documentation files in all development efforts
B. Testing the stress levels of developer team members for all project stakeholders
C. Testing the system beyond normal operational capacity to find its breaking point
D. Testing with easy and simple valid input data only within the given constraints

Answer: C

Q577. What is test-driven development (TDD)?

A. Writing tests before writing the code to make them pass
B. Performing testing without any source code at all
C. Performing all testing after coding is fully done
D. Developing test plans and nothing else at all

Answer: A

Q578. What is a test stub?

A. A test case that has previously failed and broke and related components
B. A very short and brief test case scenario only by the project team members
C. A comprehensive test report and document summary to achieve project objectives
D. A simplified implementation that provides predetermined responses during testing

Answer: D

Q579. What is a mock object?

A. A specific type of database storage system type as part of the methodology
B. A decorative object for the office workspace area for effective project outcomes
C. A simulated object that mimics the behavior of real objects in a controlled way for testing
D. A counterfeit fake commercial product item only in the engineering discipline

Answer: C

Q580. What is alpha testing vs beta testing?

A. Alpha is internal testing by developers; beta is external testing by end users
B. Alpha testing is always done by external end users within the project scope
C. The two types of testing are the same thing exactly in the development process
D. Beta testing always comes before alpha testing does during the software lifecycle

Answer: A

Q581. What is Lehman's Law of Increasing Complexity?

A. As a system evolves, its complexity increases unless work is done to reduce it
B. Complexity is completely irrelevant to all quality over the entire lifecycle
C. Complexity always stays perfectly constant overall and its related activities
D. Complexity always decreases over time by itself by the development process

Answer: A

Q582. What is reverse engineering in software maintenance?

A. Analyzing existing software to understand its design and implementation
B. Completely uninstalling the software product system within the given constraints
C. Writing application source code backwards manually for all project stakeholders
D. Building and manufacturing physical cars and parts as defined by standards

Answer: A

Q583. What is re-engineering in software?

A. Deploying software to production server systems now and related components
B. Restructuring or rewriting existing software to improve its quality without changing functionality
C. Permanently deleting the software product entirely by the project team members
D. Starting completely over from scratch and beginning in all development efforts

Answer: B

Q584. What is the impact analysis in maintenance?

A. Conducting performance benchmarking analysis tests
B. Analyzing user behavior and usage pattern metrics
C. Performing financial analysis of the project budget
D. Assessing the effects of proposed changes on the rest of the system

Answer: D

Q585. What is software aging?

A. Developer aging and gaining experience over career for effective project outcomes
B. Physical hardware aging over many years of running in the engineering discipline
C. The degradation of software quality over time due to accumulated changes and environmental evolution
D. Software physically getting old and worn down over time to achieve project objectives

Answer: C

Q586. What is the difference between maintenance and evolution?

A. Maintenance is always more comprehensive overall during the software lifecycle
B. Maintenance fixes and adapts; evolution is the broader process of continuous change
C. The two concepts are completely identical overall as part of the methodology
D. Evolution is just a type of maintenance activity in the development process

Answer: B

Q587. What is program comprehension?

A. Reading a technical programming textbook chapter
B. Understanding existing code to perform maintenance or modifications
C. Compiling source code into executable file formats
D. Writing brand new application source code logic

Answer: B

Q588. What is a maintenance request?

A. A physical hardware equipment purchase request doc
B. A purchase order for new office supply materials
C. A formal request to change, fix, or enhance existing software
D. A personal vacation and time off request document

Answer: C

Q589. What is the concept of 'software entropy'?

A. The energy consumption of running software systems for the development team
B. The tendency of software to become more disordered and difficult to maintain over time
C. A concept from physics thermodynamics discipline within the project scope
D. An established coding standard and guideline type throughout the project

Answer: B

Q590. What is migration in software maintenance?

A. Moving to a new country or city location abroad in the system context
B. Moving software from one environment, platform, or technology to another
C. Installing new physical hardware equipment devices for quality purposes
D. Permanently deleting old legacy software and files during implementation

Answer: B

Q591. What is the difference between a walkthrough and an inspection?

A. A walkthrough always has defined formal roles setup in all development efforts
B. A walkthrough is informal and author-led; an inspection is formal with defined roles
C. An inspection is always informal and casual in tone within the given constraints
D. The two approaches are completely identical overall for all project stakeholders

Answer: B

Q592. What is the ISO/IEC 25010 quality model?

A. A standard defining software quality characteristics like functionality, reliability, and usability
B. A relational database normalization model and form in the engineering discipline
C. A specific automobile car model type and category by the project team members
D. A software architecture design model pattern type and related components to achieve project objectives

Answer: A

Q593. What is formal technical review (FTR)?

A. A regular source code coding session for features as part of the methodology
B. A structured review technique with specific roles and procedures to find defects
C. An informal casual team meeting with no structure for effective project outcomes
D. A production deployment meeting and schedule plan in the development process

Answer: B

Q594. What is the defect density metric?

A. Number of defects per individual developer person during the software lifecycle
B. The number of defects found per unit size of software (e.g., per KLOC)
C. The severity rating of individual defects and bugs for the development team
D. The total number of all defects combined overall within the project scope

Answer: B

Q595. What is the concept of 'shift left' in quality?

A. Moving the computer screen display to the left side throughout the project
B. Moving deployment earlier in the schedule timeline during implementation
C. Moving testing and quality activities earlier in the development lifecycle
D. Shifting team members to different project assignments in the system context

Answer: C

Q596. What is a quality audit?

A. An independent evaluation of software processes and products against standards
B. A financial accounting audit process only exclusively for quality purposes
C. A system performance load test execution and report as a standard approach
D. A formal source code peer review session and meeting in practice typically

Answer: A

Q597. What is the Pareto principle in QA?

A. All modules have an equal number of bugs and issues
B. Equal distribution of bugs across all code modules
C. Approximately 80% of defects come from 20% of the modules
D. Testing all modules equally and consistently overall

Answer: C

Q598. What is defect prevention?

A. Activities aimed at identifying and addressing the root causes of defects before they occur
B. Testing only at the very end of the whole project in a systematic way
C. Finding defects only after production release launch at every stage across all phases
D. Deliberately ignoring all known defects and issues by the organization

Answer: A

Q599. What is the role of a QA engineer?

A. Ensuring software quality through planning, testing, and process improvement
B. Only writing project documentation materials only within the system boundary
C. Only writing application source code exclusively for the project goals
D. Only managing projects and schedules exclusively according to best practices

Answer: A

Q600. What are the internal and external failure costs?

A. Internal failure costs are always much higher overall for all project stakeholders within the given constraints
B. External failure costs are always much cheaper overall over the entire lifecycle as defined by standards
C. The two cost types are fundamentally the same thing by the development process and its related activities
D. Internal failures are found before release (rework); external failures are found after release (support, reputation)

Answer: D

Q601. What is cyclomatic complexity?

A. A complexity metric related to bicycles and riding in practice typically
B. The total count of all declared variable name types by the organization
C. The total length of all source code files and lines as a standard approach
D. A metric measuring the number of linearly independent paths through a program's source code

Answer: D

Q602. What is function point analysis?

A. Counting total functions in source code and files at every stage
B. A method of measuring software size based on the functionality delivered to users
C. A specific software testing technique approach only across all phases
D. An established coding standard and guideline set only in a systematic way

Answer: B

Q603. What is the Halstead complexity metric?

A. A set of metrics based on counting operators and operands in source code
B. A comprehensive software testing metric set and type according to best practices
C. A subjective difficulty rating for the code base for the project goals
D. A software architecture design metric set and type within the system boundary

Answer: A

Q604. What is the difference between direct and indirect metrics?

A. Indirect metrics are always more accurate and precise over the entire lifecycle as defined by standards
B. Direct metrics are always calculated from other data for all project stakeholders
C. Direct metrics are measured directly (LOC); indirect metrics are derived from other measures (productivity)
D. The two metric types are completely the same thing by the development process and its related activities

Answer: C

Q605. What is the maintainability index?

A. A comprehensive software testing score and rating
B. An index at the back of a textbook publication
C. A production deployment readiness rating and score
D. A composite metric indicating how easy it is to maintain software

Answer: D

Q606. What is cohesion metric?

A. A coupling metric between different system modules in all development efforts
B. A comprehensive software testing metric and score by the project team members
C. A measure of how strongly related the responsibilities within a module are
D. A measure of team togetherness and morale level within the given constraints

Answer: C

Q607. What is coupling metric?

A. A production deployment readiness metric and score
B. A measure of how strongly one module depends on other modules
C. A comprehensive software testing metric and score
D. A cohesion metric within a single code module only

Answer: B

Q608. What is the fan-in and fan-out metric?

A. A cooling system performance metric for hardware and related components
B. A relational database performance metric and score in the engineering discipline
C. A network bandwidth performance metric measurement to achieve project objectives
D. Fan-in counts modules calling a module; fan-out counts modules called by a module

Answer: D

Q609. What is the depth of inheritance tree (DIT) metric?

A. The depth of a relational database schema and tables as part of the methodology
B. The height of a tree in a garden setting outdoors for effective project outcomes
C. The depth of software testing coverage and breadth in the development process
D. The maximum length of the path from a class to the root class in an inheritance hierarchy

Answer: D

Q610. What is the weighted methods per class (WMC) metric?

A. The total number of classes in the project source
B. The total number of source code files in project
C. The average physical weight of all methods defined
D. The sum of complexities of all methods defined in a class

Answer: D

Q611. What is the purpose of a Change Control Board (CCB)?

A. A physical whiteboard in the office space and room by the organization
B. A dedicated software testing team or group of people at every stage
C. A development source coding team or group of people across all phases
D. A group responsible for evaluating and approving or rejecting change requests

Answer: D

Q612. What is a merge conflict?

A. A situation where two branches have competing changes that cannot be automatically merged
B. A software testing failure or assertion error result according to best practices
C. A disagreement or argument between the team members in a systematic way
D. A production deployment failure or error issue event for the project goals

Answer: A

Q613. What is continuous integration in the context of SCM?

A. Frequently merging code changes to a shared repository with automated builds and tests
B. Manually merging code without any automation tools by the development process
C. Performing one-time integration at the very end and its related activities
D. Performing annual scheduled integration events only within the system boundary

Answer: A

Q614. What is a tag in version control?

A. A price tag attached to a retail product for sale over the entire lifecycle
B. A commit message attached to a changeset in version control for all project stakeholders
C. A marker that identifies a specific point in the repository's history, typically a release
D. A specific type of development branch in the project as defined by standards

Answer: C

Q615. What is the difference between centralized and distributed version control?

A. Centralized version control is the newer approach today by the project team members
B. Distributed version control has only one server overall in all development efforts
C. The two systems are completely identical in every way within the given constraints
D. Centralized has one server; distributed gives each developer a full copy of the repository

Answer: D

Q616. What is a build system in configuration management?

A. A physical construction building system and process and related components
B. A project documentation management system and process in the engineering discipline
C. A tool that automates the process of compiling source code into executable software
D. A comprehensive software testing system and tool suite to achieve project objectives

Answer: C

Q617. What is release management?

A. A comprehensive software testing method and approach type in the development process
B. Releasing employees from their positions and roles for effective project outcomes
C. A source code implementation coding technique and method as part of the methodology
D. The process of planning, scheduling, and controlling software builds through different stages

Answer: D

Q618. What is a software configuration audit?

A. A financial accounting audit process and procedure only during the software lifecycle
B. A system performance load test execution and run report for the development team
C. A formal source code peer review session and meeting only within the project scope
D. A review to verify that configuration items conform to specifications and standards

Answer: D

Q619. What is GitFlow?

A. A software testing flow and process model and approach in the system context
B. A water flow through pipes and channels and drains throughout the project
C. A production deployment pipeline and flow and process during implementation
D. A branching model for Git that defines a strict branching structure for projects

Answer: D

Q620. What is trunk-based development?

A. A branching strategy where developers merge small, frequent updates to a core trunk or main branch
B. Developing software in physical tree trunks outdoors for quality purposes in practice typically
C. A comprehensive software testing approach and method only as a standard approach
D. A software architecture and design method approach only by the organization

Answer: A

Q621. What is a risk exposure (risk magnitude)?

A. A production deployment issue and concern problem
B. A type of security vulnerability and exposure found
C. The product of risk probability and risk impact
D. Physical exposure to sunlight and weather outdoors

Answer: C

Q622. What is the difference between known risks and unknown risks?

A. Known risks cannot be managed effectively at all within the given constraints
B. Unknown risks are far more common than known risks for all project stakeholders
C. Known risks can be identified and planned for; unknown risks cannot be anticipated
D. The two categories are fundamentally the same thing as defined by standards

Answer: C

Q623. What is Boehm's risk management approach?

A. A production deployment automation approach and plan
B. A framework with two main components: risk assessment and risk control
C. A source code implementation coding approach and plan
D. A comprehensive software testing approach and process

Answer: B

Q624. What is the RMMM plan?

A. A comprehensive software testing plan and document only and related components
B. Risk Mitigation, Monitoring, and Management Plan — a document for managing project risks
C. A source code implementation coding plan and document by the project team members
D. A plan based on Roman numeral notation and numbering in all development efforts

Answer: B

Q625. What is a contingency plan?

A. The main and primary project plan document overall
B. A production deployment plan and schedule doc
C. A comprehensive software testing plan document
D. A backup plan activated when a risk materializes

Answer: D

Q626. What are the top 10 software risks according to Boehm?

A. Ten common source code coding errors and mistakes to achieve project objectives
B. Ten common software testing risk types and categories in the engineering discipline
C. Ten common deployment issues and concerns and problems for effective project outcomes
D. Risks including personnel shortfalls, unrealistic schedules, wrong requirements, and gold plating

Answer: D

Q627. What is risk monitoring?

A. Continuously tracking identified risks and identifying new risks throughout the project
B. Monitoring application source code changes and edits as part of the methodology
C. Monitoring production server health status and uptime in the development process
D. Monitoring individual team member performance metrics during the software lifecycle

Answer: A

Q628. What is a risk trigger?

A. A physical gun trigger mechanism device component only
B. A software testing trigger or activation event and signal
C. An indicator or symptom that a risk event is about to occur or has occurred
D. A production deployment trigger or activation signal only

Answer: C

Q629. What is the difference between qualitative and quantitative risk analysis?

A. Quantitative analysis is always purely subjective in nature for the development team
B. The two approaches are completely identical in all ways within the project scope
C. Qualitative analysis always uses only numeric data values throughout the project
D. Qualitative uses subjective assessment (high/medium/low); quantitative uses numerical probabilities

Answer: D

Q630. What is a risk breakdown structure?

A. A comprehensive software testing structure and model for quality purposes
B. A source code structure and organization model layout during implementation
C. A broken system that needs repair work and fixing in the system context
D. A hierarchical organization of identified risks categorized by risk source

Answer: D

Q631. What is the OWASP Top 10?

A. A popular music chart and ranking system and listing
B. A list of the ten most critical web application security risks
C. An established coding standard and guideline and rule
D. A comprehensive software testing framework tool and set

Answer: B

Q632. What is the principle of least privilege?

A. Maximum access permissions for all system users always across all phases
B. No access permissions for any user in the system ever in a systematic way
C. Users should only have the minimum access rights necessary to perform their tasks
D. Giving everyone full administrator access to everything at every stage

Answer: C

Q633. What is a buffer overflow?

A. Having too much data stored in a buffer data area for the project goals
B. A database table overflow and storage issue and error within the system boundary
C. A vulnerability where data written beyond a buffer's boundary can overwrite adjacent memory
D. A network traffic overflow and congestion issue event according to best practices

Answer: C

Q634. What is input validation?

A. Accepting all user input without any validation checks by the development process
B. Validating only system output and response data values and its related activities
C. Testing user interface elements for usability and flow over the entire lifecycle
D. Checking and sanitizing user input to prevent malicious data from entering the system

Answer: D

Q635. What is defense in depth?

A. A security approach using multiple layers of security controls throughout a system
B. Defending only the database from attacks and threats within the given constraints
C. Having one single strong defense mechanism and barrier for all project stakeholders
D. A military strategy used only in warfare and battles as defined by standards

Answer: A

Q636. What is a security audit?

A. A financial accounting audit process and procedure type in all development efforts
B. A systematic evaluation of a system's security by examining compliance with security policies
C. A system performance load test execution and run process and related components
D. A formal source code peer review session and process only by the project team members

Answer: B

Q637. What is penetration testing?

A. Physical building break-in testing attempt and effort
B. Database query performance testing and benchmarking
C. Network communication speed testing and measurement
D. Simulating cyberattacks to identify security vulnerabilities in a system

Answer: D

Q638. What is a denial-of-service (DoS) attack?

A. A specific type of computer virus or malware and threat in the engineering discipline
B. Denying customer service requests politely and kindly to achieve project objectives
C. An attack that overwhelms a system with traffic to make it unavailable to users
D. A network configuration and setup procedure and process for effective project outcomes

Answer: C

Q639. What is the CIA triad in security?

A. A government intelligence agency abbreviation and name as part of the methodology
B. An established coding standard and guideline and rule set in the development process
C. Confidentiality, Integrity, and Availability — the three core principles of information security
D. A comprehensive software testing framework and tool suite during the software lifecycle

Answer: C

Q640. What is secure coding?

A. Writing code in a physically secure room and building
B. Encrypting all application source code files and folders
C. Development practices that protect software from security vulnerabilities
D. Using complex passwords directly in application code

Answer: C

Q641. What is the difference between verification and validation?

A. Verification writes documentation; validation writes test code
B. Verification deploys software; validation monitors production
C. Verification checks correctness; validation checks usefulness
D. Verification tests performance; validation tests user interface

Answer: C

Q642. Which factor most contributes to the software crisis?

A. Decreasing cost of modern hardware and peripherals
B. Growing availability of trained software professionals
C. Increasing system complexity and changing requirements
D. Widespread adoption of standardized coding practices

Answer: C

Q643. What is the significance of the NATO 1968 conference?

A. It launched the initial version of the world wide web
B. It formally introduced software engineering as a discipline
C. It created the first commercial operating system product
D. It established the first programming language standard

Answer: B

Q644. How does software engineering differ from programming?

A. It focuses only on writing source code in languages
B. It deals exclusively with database design and queries
C. It encompasses the entire development lifecycle process
D. It concerns itself primarily with hardware maintenance

Answer: C

Q645. What is the purpose of a feasibility study in software projects?

A. To write the final source code for the application
B. To conduct user acceptance testing after development
C. To deploy the software to the production environment
D. To determine if the project is viable and worthwhile

Answer: D

Q646. Which software characteristic ensures it works across platforms?

A. Obscurity of the source code implementation
B. Dependency on a single hardware configuration
C. Portability across different operating environments
D. Complexity of the underlying algorithm design

Answer: C

Q647. What is meant by 'technical debt' in software engineering?

A. The monetary budget allocated for purchasing hardware
B. Revenue generated from software licensing agreements
C. Total salary expenses for the development team members
D. Accumulated cost of shortcuts taken during development

Answer: D

Q648. Why are software engineering ethics considered important?

A. They guide responsible behavior in professional practice
B. They reduce the physical size of deployed software
C. They increase the execution speed of the application
D. They eliminate the need for comprehensive testing

Answer: A

Q649. What is the relationship between software and hardware engineering?

A. Both disciplines use identical processes and methodologies
B. Software engineering is a direct subset of hardware engineering
C. Hardware engineering completely depends on software practices
D. Software engineering builds on but differs from hardware methods

Answer: D

Q650. What is the primary benefit of using CASE tools?

A. They automate and support software development activities
B. They convert hardware specifications into software code
C. They eliminate all possible defects from the source code
D. They replace the need for any human software developers

Answer: A

Q651. What is the key difference between incremental and iterative models?

A. Incremental refines features; iterative adds new functionality
B. Both approaches are completely identical in every possible way
C. Neither approach involves any form of planning or documentation
D. Incremental adds functionality; iterative refines existing features

Answer: D

Q652. In the spiral model, what happens at the end of each spiral loop?

A. Testing is permanently skipped for all remaining spiral loops
B. The entire project is immediately deployed to production servers
C. A review determines whether to continue to the next iteration
D. All source code is discarded and rewritten from scratch again

Answer: C

Q653. Why might a team choose the RAD model for development?

A. When no user involvement is needed throughout development
B. When requirements are completely fixed and will never change
C. When the project has unlimited budget and timeline available
D. When rapid delivery of a working system is the top priority

Answer: D

Q654. What is a disadvantage of the waterfall model?

A. Excessive flexibility leading to uncontrolled scope expansion
B. Difficulty accommodating requirement changes after phase completion
C. Lack of any structured phases making progress hard to track
D. Too many iterative cycles causing confusion among developers

Answer: B

Q655. How does the unified process (UP) organize development activities?

A. Into four phases: inception, elaboration, construction, transition
B. Into two phases: planning and coding without any other steps
C. Into six phases that must always be executed sequentially only
D. Into a single continuous phase without any distinct milestones

Answer: A

Q656. What role does risk assessment play in the spiral model?

A. It drives decisions about which activities to perform each cycle
B. It has no significant impact on how the project is conducted
C. It replaces all other engineering activities in the entire model
D. It is performed only once at the very beginning of the project

Answer: A

Q657. When is the evolutionary development model most appropriate?

A. When initial requirements are unclear and need exploration work
B. When all requirements are perfectly defined before starting
C. When the project has a very short and fixed delivery deadline
D. When the team prefers no stakeholder interaction during work

Answer: A

Q658. What is the main criticism of the code-and-fix development approach?

A. It requires too much upfront planning and design documentation
B. It mandates excessive testing before any code is ever written
C. It leads to poorly structured software that is hard to maintain
D. It forces teams to follow overly rigid process model guidelines

Answer: C

Q659. What is the purpose of the elaboration phase in the Unified Process?

A. To deploy the completed software to end users in production now
B. To analyze the problem domain and establish architectural foundation
C. To gather initial business case and scope project requirements
D. To perform final acceptance testing before system goes live now

Answer: B

Q660. How does the component-based development model reduce development effort?

A. By eliminating all testing and quality assurance activities fully
B. By avoiding any form of architectural design or documentation
C. By reusing pre-built and tested software components extensively
D. By skipping the requirements gathering phase of development work

Answer: C

Q661. What is the difference between Scrum and Kanban?

A. Scrum avoids visualization; Kanban prohibits all meetings
B. Scrum has no backlog; Kanban requires sprint planning
C. Scrum has no roles; Kanban requires a Scrum Master role
D. Scrum uses time-boxed sprints; Kanban uses continuous flow

Answer: D

Q662. What is velocity in agile project management?

A. The frequency of deploying code to the production servers
B. The amount of work a team completes in one iteration period
C. The speed at which the application executes user requests
D. The rate at which new developers join the project team

Answer: B

Q663. What is the purpose of a sprint retrospective?

A. To create the initial project charter and scope document
B. To identify improvements for the team process going forward
C. To demonstrate completed features to external stakeholders
D. To assign blame for any defects found during the sprint

Answer: B

Q664. How does pair programming benefit agile teams?

A. It allows one developer to do all work while the other rests
B. It doubles the number of features developed per time unit
C. It increases code quality through continuous peer review work
D. It eliminates the need for any automated testing frameworks

Answer: C

Q665. What is the definition of 'done' in Scrum?

A. A legal document signed by the client at project conclusion
B. A shared understanding of when a work item is complete enough
C. A report generated automatically by the build server system
D. A metric measuring the total number of lines of code written

Answer: B

Q666. What is a burndown chart used for in agile?

A. Documenting all architectural decisions made during the project
B. Tracking remaining work against time in a sprint or release
C. Measuring the total revenue generated by the software product
D. Recording the number of team members joining or leaving work

Answer: B

Q667. What is the purpose of story points in agile estimation?

A. To measure the physical size of the application in megabytes
B. To estimate the relative effort required for completing tasks
C. To count the exact number of hours each task will take to do
D. To track the number of bugs reported by end users each week

Answer: B

Q668. How does continuous integration support agile development?

A. By eliminating the need for version control systems entirely
B. By delaying all testing until the final sprint of the project
C. By preventing developers from committing code to the repository
D. By frequently merging and testing code to detect issues early

Answer: D

Q669. What is the role of acceptance criteria in user stories?

A. They list the hardware requirements for running the software
B. They specify the programming language to be used for coding
C. They define conditions that must be met for story completion
D. They document the salary expectations of the development team

Answer: C

Q670. What is a spike in agile development?

A. A formal review process conducted by external quality auditors
B. A sudden increase in the number of production bugs per release
C. A time-boxed research task to reduce uncertainty or risk factors
D. A mandatory overtime period for all team members on a project

Answer: C

Q671. What is the difference between user requirements and system requirements?

A. User requirements are high-level needs; system requirements are detailed specifications
B. User requirements are optional; system requirements are always mandatory for the project
C. User requirements come from developers; system requirements come from end user groups
D. User requirements are written in code; system requirements are written in plain language

Answer: A

Q672. What is requirements prioritization and why is it necessary?

A. It ranks requirements by importance to guide development order and resource allocation
B. It randomly assigns requirements to different sprints without any logical ordering method
C. It permanently removes low-priority requirements from the project scope and backlog
D. It ensures all requirements are implemented simultaneously in a single development phase

Answer: A

Q673. What challenges arise during requirements elicitation with multiple stakeholders?

A. All stakeholders always agree perfectly on every requirement without any discussion
B. Multiple stakeholders simplify the process by reducing the total number of requirements
C. Having more stakeholders eliminates the need for formal requirements documentation
D. Conflicting viewpoints and priorities that must be negotiated and resolved carefully

Answer: D

Q674. How do prototypes help in requirements engineering?

A. They replace the need for any written requirements documentation in the project
B. They help stakeholders visualize the system and clarify unclear requirements early
C. They guarantee that all requirements are correctly implemented in the final system
D. They eliminate the possibility of any requirements changes during the development

Answer: B

Q675. What is the MoSCoW method in requirements prioritization?

A. A geographic approach assigning requirements to teams based on physical location
B. A technique categorizing requirements as Must, Should, Could, and Won't have items
C. A testing strategy for validating requirements against acceptance criteria only
D. A programming paradigm for implementing requirements in object-oriented languages

Answer: B

Q676. What is the difference between requirements verification and validation?

A. Verification writes test cases; validation executes marketing campaigns for launch
B. Verification checks document consistency; validation checks stakeholder satisfaction
C. Verification trains users; validation calculates the return on investment figures
D. Verification deploys the system; validation monitors production performance metrics

Answer: B

Q677. Why is natural language problematic for writing requirements?

A. It is too precise and does not allow any flexibility in requirement statements
B. It is only understood by technical staff and not by any business stakeholders
C. It is inherently ambiguous and can lead to multiple conflicting interpretations
D. It cannot express complex requirements due to limited vocabulary available

Answer: C

Q678. What is a requirements review and who should participate?

A. A formal inspection of requirements involving stakeholders and technical staff
B. A marketing meeting to discuss product positioning and pricing strategies
C. An informal chat between two developers about their coding preferences only
D. A hardware procurement session to select servers for the production use

Answer: A

Q679. How does requirements management handle change requests?

A. By delegating all change decisions to a single junior developer on the team
B. By rejecting all change requests to maintain the original project scope fully
C. By immediately implementing all requested changes without any analysis first
D. Through a formal change control process that evaluates impact before approval

Answer: D

Q680. What is the role of domain analysis in requirements engineering?

A. It focuses exclusively on selecting the programming language for development
B. It calculates the exact project budget and timeline before development starts
C. It determines the number of servers needed for production deployment only
D. It helps understand the application domain to identify relevant requirements

Answer: D

Q681. What is the purpose of earned value management (EVM)?

A. To measure project performance by comparing planned versus actual progress and costs
B. To determine which programming language is most suitable for the project requirements
C. To calculate the total number of lines of code written by each developer on the team
D. To design the system architecture and select appropriate technology stack components

Answer: A

Q682. How does the critical path method (CPM) help project managers?

A. It guarantees that no project will ever experience delays or budget overruns
B. It identifies tasks that directly impact project completion date if delayed at all
C. It randomly assigns priorities to tasks without considering their dependencies now
D. It eliminates the need for any project scheduling or timeline management work

Answer: B

Q683. What is scope creep and how can it be managed?

A. An issue that resolves itself automatically without any management intervention
B. A beneficial practice that should be encouraged for maximum project flexibility
C. A problem that only affects projects using agile development methodologies now
D. Uncontrolled scope expansion managed through formal change control processes

Answer: D

Q684. What is the difference between effort estimation and duration estimation?

A. Effort measures code quality; duration measures the number of features implemented
B. Effort is only relevant for testing; duration is only relevant for coding activities
C. Effort is total work hours needed; duration is calendar time to complete the task
D. Effort and duration are identical concepts with no meaningful difference between them

Answer: C

Q685. What is the COCOMO model used for in software project management?

A. Managing source code versions and branches in the project code repository
B. Testing software applications for security vulnerabilities and performance issues
C. Estimating software development effort and cost based on project size parameters
D. Designing user interfaces for web-based applications and mobile platforms only

Answer: C

Q686. How do project managers handle team conflicts effectively?

A. By immediately removing the conflicting team members from the project team
B. By assigning all disputed tasks to a single person to avoid disagreements
C. By facilitating open communication and finding mutually acceptable solutions
D. By ignoring conflicts and hoping they resolve themselves without intervention

Answer: C

Q687. What is a risk register in project management?

A. A source code repository containing all versions of the application being built
B. A database of all employee records including salary and performance evaluations
C. A financial ledger tracking all project expenses and revenue over the timeline
D. A documented list of identified risks with their probability and impact assessments

Answer: D

Q688. What is the purpose of a project post-mortem or lessons learned session?

A. To calculate the final salary bonuses for the project team members after launch
B. To assign blame to team members who made mistakes during the project execution
C. To identify what went well and what can be improved for future project efforts
D. To permanently archive all project documents and prevent any future access now

Answer: C

Q689. How does function point analysis help in project estimation?

A. It measures the physical size of the deployed application in storage megabytes
B. It tracks the number of team members assigned to the project at any given time
C. It counts the number of programming languages used in the project source code
D. It measures software size based on functional requirements independent of technology

Answer: D

Q690. What is the triple constraint in project management?

A. The balance between scope, time, and cost that defines project boundaries
B. The three phases of testing: unit, integration, and system level testing
C. The minimum number of developers needed to start any software project
D. The three programming languages required for every software project build

Answer: A

Q691. What is the Single Responsibility Principle (SRP)?

A. A class should handle all possible system responsibilities now
B. A function should always return multiple values simultaneously
C. A class should have only one reason to change its behavior
D. A module should depend on as many other modules as possible

Answer: C

Q692. What is the Open-Closed Principle in software design?

A. Software entities should never be extended or modified after initial release
B. Software entities should be open for extension but closed for modification
C. Software entities should be both open and closed for all types of changes
D. Software entities should be closed for extension and open for modification

Answer: B

Q693. What is the difference between the Strategy and State design patterns?

A. Strategy and State are identical patterns with no differences between them at all
B. Strategy selects algorithms; State changes behavior based on internal object state
C. Strategy manages object creation; State handles object destruction and cleanup work
D. Strategy is for UI design only; State is exclusively for database operations work

Answer: B

Q694. How does the Observer pattern support loose coupling?

A. Observers must inherit from the subject class to receive any notification updates
B. Observers must know the complete internal state of the subject at all times now
C. Subjects notify observers without knowing their concrete implementation details
D. The pattern requires tight coupling between all observers and the subject object

Answer: C

Q695. What is the Dependency Inversion Principle?

A. High-level modules should depend on abstractions not on low-level module details
B. Dependencies should always flow from abstractions down to concrete implementations
C. High-level modules should directly depend on the implementation of low-level modules
D. Low-level modules should control the behavior of all high-level modules in system

Answer: A

Q696. What is the Factory Method pattern used for?

A. Connecting to external databases and executing queries against stored data records
B. Sorting collections of objects based on their attributes in ascending or descending order
C. Destroying objects and freeing memory after they are no longer needed by the program
D. Creating objects without specifying the exact class to instantiate at compile time

Answer: D

Q697. How does the MVC pattern separate concerns in application design?

A. It eliminates the need for any user interface by focusing solely on data storage
B. It only separates the database from the rest of the application into two layers
C. It separates data (Model), presentation (View), and logic (Controller) into layers
D. It combines all application logic into a single layer without any separation at all

Answer: C

Q698. What is the Liskov Substitution Principle?

A. Base types should be substitutable for their subtypes in all program situations
B. Subtypes should completely change the behavior defined by their base type classes
C. Subtypes must be substitutable for their base types without altering correctness
D. Inheritance should be avoided entirely in favor of composition in every case

Answer: C

Q699. What is the purpose of the Adapter design pattern?

A. To make incompatible interfaces work together through a wrapper component layer
B. To add new functionality to objects dynamically at runtime without inheritance use
C. To ensure a class has only one instance throughout the entire application system
D. To create new objects by cloning existing ones without using constructors directly

Answer: A

Q700. What is the difference between composition and inheritance in design?

A. Composition is always inferior to inheritance for all software design scenarios today
B. Composition uses has-a relationships; inheritance uses is-a relationships between types
C. Composition and inheritance are identical concepts with no meaningful differences at all
D. Inheritance provides more flexibility than composition in every possible design context

Answer: B

Q701. What is the difference between software architecture and software design?

A. Architecture and design are identical concepts with no meaningful differences between them
B. Architecture focuses on code details; design focuses on system-level structural decisions
C. Architecture defines high-level structure; design details component internals and algorithms
D. Architecture is only relevant for small systems; design is only for large-scale systems

Answer: C

Q702. How does the Model-View-Controller architecture support separation of concerns?

A. It eliminates the need for a user interface by focusing exclusively on data processing
B. It only separates the database from the application without any further decomposition
C. It separates data management, user interface, and control logic into distinct components
D. It combines all concerns into a single component for simplicity and ease of maintenance

Answer: C

Q703. What are the trade-offs of microservices versus monolithic architecture?

A. Microservices offer scalability and independence but add complexity in communication and deployment
B. There are no meaningful trade-offs because both approaches produce identical system qualities
C. Microservices are always superior to monolithic architectures in every possible project scenario
D. Monolithic architectures are always more scalable than microservices in all system contexts today

Answer: A

Q704. What is an event-driven architecture?

A. Components communicate by producing and consuming events asynchronously via message brokers
B. All system events are processed in a strictly sequential order without any parallel execution
C. Components communicate exclusively through synchronous direct function calls without any delay
D. Event-driven systems cannot handle more than one event at any given point in time today

Answer: A

Q705. How does the service-oriented architecture (SOA) promote reusability?

A. Services are designed as reusable, self-contained units with well-defined standard interfaces
B. SOA discourages reusability by requiring unique implementations for every system integration
C. Services in SOA are always built from scratch and never leverage any existing functionality
D. Services are tightly coupled and cannot be reused outside their original application context

Answer: A

Q706. What is an architectural decision record (ADR)?

A. A project management artifact tracking team member assignments and work allocations
B. A source code file containing the main application entry point and initialization logic
C. A database schema definition file used for creating tables and relationships in storage
D. A document capturing the context, decision, and consequences of an architectural choice

Answer: D

Q707. What is the strangler fig pattern in architecture migration?

A. Running both old and new systems permanently without ever completing the migration work
B. Immediately shutting down the old system and deploying the new one all at once together
C. Abandoning the migration and keeping the legacy system unchanged for all future operation
D. Gradually replacing parts of a legacy system with new components until fully migrated

Answer: D

Q708. How does a message queue improve system architecture?

A. It tightly couples all components requiring synchronous communication between every service
B. It decouples producers and consumers allowing asynchronous communication and load leveling
C. It reduces system reliability by introducing additional points of failure without any benefit
D. It eliminates the need for any communication between system components or services at all

Answer: B

Q709. What is the CQRS (Command Query Responsibility Segregation) pattern?

A. Separating read and write operations into different models for optimization flexibility
B. Using the same data model and access path for both queries and commands in all cases
C. Combining all read and write operations into a single model for maximum simplicity only
D. Eliminating all read operations and only allowing write operations in the system design

Answer: A

Q710. What is the role of middleware in a distributed architecture?

A. It only stores data and has no role in communication or transformation between services
B. It provides communication, data transformation, and integration between distributed components
C. It eliminates the need for any networking between distributed system components entirely
D. It replaces all business logic components and handles every processing requirement directly

Answer: B

Q711. What are Nielsen's ten usability heuristics used for?

A. Evaluating user interface designs for common usability problems and issues
B. Configuring network firewalls for protecting against security threats
C. Designing database schemas for storing application data efficiently now
D. Writing backend server code for processing user authentication requests

Answer: A

Q712. What is the difference between UI design and UX design?

A. UI and UX are identical disciplines with no differences in scope or focus areas
B. UI covers project management; UX handles server infrastructure configuration
C. UI focuses on visual elements; UX encompasses the entire user experience journey
D. UI deals with backend logic; UX focuses exclusively on database design work now

Answer: C

Q713. What is Fitts' law and how does it apply to UI design?

A. Smaller and distant targets are easier to click, so buttons should be minimized always
B. Fitts' law only applies to physical devices and has no relevance to digital interfaces
C. Larger and closer targets are faster to select, influencing button size and placement
D. Target size has no impact on user interaction speed or accuracy in any interface now

Answer: C

Q714. How does the principle of consistency improve user interfaces?

A. It makes the interface unpredictable so users must think carefully about every action
B. It allows users to apply learned behaviors across the interface reducing cognitive load
C. It forces every page to look completely different from all other pages in the system
D. It increases the time users need to learn how to navigate and use the application

Answer: B

Q715. What is a design system in the context of UI development?

A. A single page design template that cannot be customized or reused across the application
B. A collection of reusable components and guidelines ensuring visual and behavioral consistency
C. A database management system for storing user interface configuration data and settings
D. A server monitoring tool for tracking application performance metrics and error rates

Answer: B

Q716. What is the purpose of A/B testing in UI design?

A. Simultaneously releasing both design variants as permanent features for all users
B. Testing the application's backend server performance under heavy load conditions
C. Validating the database schema against the documented requirements specification
D. Comparing two design variants to determine which performs better with real users

Answer: D

Q717. How does progressive disclosure improve complex user interfaces?

A. It reveals information gradually, showing only what is needed at each step of interaction
B. It hides all information permanently and never reveals anything to the user at any time
C. It displays all possible options and information simultaneously on a single crowded screen
D. It forces users to read complete documentation before they can interact with the system

Answer: A

Q718. What is the importance of color contrast in UI design?

A. It ensures text and elements are readable for users including those with visual impairments
B. High contrast should be avoided because it makes interfaces harder for everyone to read
C. Color contrast only matters for printed materials and is irrelevant for digital interfaces
D. It is purely decorative and has no impact on readability or accessibility of the interface

Answer: A

Q719. What is a persona in user-centered design?

A. A software bot that automatically generates user interface layouts without human input
B. A legal document defining the terms of service between the company and its end users
C. A real person hired to test the application before it is released to the general public
D. A fictional character representing a target user group based on research data analysis

Answer: D

Q720. What is the purpose of user journey mapping in UI design?

A. To create the database schema for storing all application data and configurations
B. To write the server-side code for processing API requests from the client side
C. To configure the deployment pipeline for releasing the application to production
D. To visualize the complete user experience across all touchpoints and interactions

Answer: D

Q721. What is the difference between compiled and interpreted languages?

A. Compiled languages run slower than interpreted languages in all performance benchmarks and tests
B. Compiled languages translate code before execution; interpreted languages translate during execution
C. Compiled and interpreted languages are identical with no differences in execution approach at all
D. Interpreted languages cannot be used for building any production-grade software applications

Answer: B

Q722. What is continuous integration (CI) in software development?

A. Manually combining all code at the end of the project without any automated testing or builds
B. Writing all source code in a single file to avoid any merge conflicts between team members
C. Deploying code directly to production without any testing or review processes being performed
D. Frequently merging code changes into a shared repository with automated build and test validation

Answer: D

Q723. How does pair programming improve code quality?

A. It reduces code quality because two developers disagree and produce inconsistent solutions
B. It doubles development speed by having two people type simultaneously on the same keyboard
C. It eliminates the need for any code reviews since the code is already reviewed during writing
D. Real-time review catches errors immediately and promotes knowledge sharing between developers

Answer: D

Q724. What is technical debt and how does it affect implementation?

A. Accumulated shortcuts requiring future rework that slows down development and increases costs
B. A financial investment made by the company to purchase better development tools and hardware
C. A type of software license that requires payment for each line of code written in the project
D. A project management metric measuring the total budget remaining for the development effort

Answer: A

Q725. What is the purpose of unit testing during implementation?

A. To check the project schedule against the planned milestones and delivery date targets
B. To test the entire application system from end to end including all integrated components
C. To validate the user interface design against the approved wireframes and mockup designs
D. To verify individual code components work correctly in isolation from other system parts

Answer: D

Q726. What is dependency injection and why is it useful?

A. Providing dependencies externally rather than creating them internally for better testability
B. Creating all dependencies inside the class constructor to simplify initialization logic now
C. Injecting database records directly into the application without using any query language
D. Removing all dependencies from a class so it operates completely independently of others

Answer: A

Q727. What is the purpose of a build automation tool?

A. To create project management reports showing team productivity and budget utilization data
B. To manually compile each source code file individually without any automated processes at all
C. To design user interfaces by automatically generating HTML and CSS from wireframe images
D. To automate compilation, testing, packaging, and deployment of software applications reliably

Answer: D

Q728. How does static code analysis help during implementation?

A. It analyzes the application user interface for usability problems and accessibility violations
B. It replaces the need for any manual code review by automatically fixing all detected issues
C. It identifies potential bugs, security vulnerabilities, and style violations without running code
D. It only detects errors that occur during runtime execution of the application in production

Answer: C

Q729. What is the difference between a framework and a library?

A. Frameworks are always superior to libraries for all software development projects and tasks
B. Frameworks and libraries are identical concepts with no meaningful differences between them
C. A framework controls the flow and calls your code; a library is called by your code directly
D. A library controls application flow; a framework provides utility functions called by code

Answer: C

Q730. What is containerization and how does it benefit implementation?

A. Packaging applications with dependencies into isolated containers ensuring consistent environments
B. Running applications directly on the host operating system without any isolation or packaging
C. Manually configuring each server individually to match the development environment settings
D. Removing all application dependencies to reduce the size of the deployed software package

Answer: A

Q731. What is the difference between black-box and white-box testing?

A. Black-box testing is always superior to white-box testing for finding all types of software bugs
B. Black-box tests functionality without code knowledge; white-box tests with code structure knowledge
C. Black-box and white-box testing are identical approaches with no meaningful differences at all
D. Black-box testing examines code structure; white-box testing ignores internal implementation fully

Answer: B

Q732. What is boundary value analysis in software testing?

A. Avoiding all edge cases and testing only the most common inputs that users are expected to use
B. Testing only the maximum possible input value and ignoring all other boundary conditions now
C. Testing at the edges of input ranges where defects are most likely to occur in the software
D. Testing random values from the middle of input ranges without focusing on any specific values

Answer: C

Q733. What is equivalence partitioning?

A. Testing every possible input value individually to ensure complete coverage of the system
B. Randomly selecting test inputs without any systematic approach to input space partitioning
C. Removing all invalid inputs from the test suite to focus only on valid expected scenarios
D. Dividing inputs into groups where all values in a group are expected to behave identically

Answer: D

Q734. What is code coverage and why does it matter?

A. It measures the percentage of code exercised by tests indicating thoroughness of test suite
B. It counts the total number of lines of code in the project regardless of testing activities
C. It measures the speed at which the application executes under normal operating conditions
D. It tracks the number of developers who have reviewed each source code file in the project

Answer: A

Q735. What is mutation testing?

A. Changing the programming language of the application to improve its overall test coverage
B. Introducing small changes to code to check if tests can detect the introduced modifications
C. Modifying the project requirements to make them easier to test with existing test suites
D. Deleting all test cases and relying on manual testing for all quality assurance activities

Answer: B

Q736. What is performance testing?

A. Checking the visual design of the user interface against approved wireframes and mockups
B. Validating that the project schedule aligns with the planned milestones and target dates
C. Verifying that individual functions return correct results for all valid input combinations
D. Evaluating system behavior under various load conditions to identify bottlenecks and limits

Answer: D

Q737. What is the purpose of test-driven development (TDD)?

A. Writing tests before code to drive design and ensure comprehensive test coverage from start
B. Using test results to determine developer salary bonuses at the end of each project phase
C. Eliminating all testing activities to focus developer time on feature implementation only
D. Writing code first and then deciding which tests to write based on the bugs found later

Answer: A

Q738. What is exploratory testing?

A. Simultaneous test design, execution, and learning without predefined scripts or test cases
B. Following a strictly predefined test script without any deviation or creativity in approach
C. Running only automated tests without any human involvement in the testing process at all
D. Exploring the source code without executing the application to find potential defect areas

Answer: A

Q739. What is the testing pyramid concept?

A. Only unit tests are needed and integration and end-to-end tests should be completely avoided
B. More end-to-end tests at the base and fewer unit tests at the top of the testing pyramid
C. More unit tests at the base, fewer integration tests, and fewest end-to-end tests at the top
D. Equal numbers of unit, integration, and end-to-end tests at every level of the testing pyramid

Answer: C

Q740. What is the difference between verification and validation in testing?

A. Verification and validation are identical activities with no differences in their purposes
B. Verification focuses on user satisfaction; validation focuses on technical correctness only
C. Verification checks if built correctly; validation checks if the right product was built
D. Verification is done after deployment; validation is done before development starts only

Answer: C

Q741. What is Lehman's law of increasing complexity?

A. Software complexity naturally decreases over time without any deliberate effort being made
B. Software complexity increases with each change unless work is done to reduce it actively
C. Software complexity remains constant throughout the entire lifetime of the application now
D. Software complexity is only affected by the initial design and never by later modifications

Answer: B

Q742. How does software aging affect system reliability over time?

A. Aging has no measurable effect on software reliability because code does not physically degrade
B. Software reliability improves automatically as hardware becomes faster and more powerful now
C. Software always becomes more reliable over time as more bugs are found and fixed in system
D. Accumulated changes and environmental drift gradually degrade system reliability and performance

Answer: D

Q743. What is the difference between re-engineering and reverse engineering?

A. Re-engineering recovers design knowledge; reverse engineering restructures for improvements now
B. Reverse engineering and re-engineering are identical processes with no differences between them
C. Reverse engineering adds new features; re-engineering only fixes bugs in the existing codebase
D. Reverse engineering recovers design; re-engineering restructures the system for improvement work

Answer: D

Q744. What is the impact of code smells on software maintenance effort?

A. Code smells always improve code quality by making the codebase more interesting to read now
B. Code smells indicate design problems that increase maintenance difficulty and defect likelihood
C. Code smells have no impact on maintenance because they do not represent actual defects today
D. Code smells are only cosmetic issues that never affect functionality or maintenance costs

Answer: B

Q745. How does modular design facilitate software maintenance?

A. Modular design makes maintenance harder because changes must be applied to every module now
B. Modular design has no relationship to maintenance effort or quality of the maintenance work
C. Modules prevent any changes from being made to the software after initial deployment to users
D. Changes can be isolated to specific modules without affecting the rest of the system at all

Answer: D

Q746. What is the ripple effect in software maintenance?

A. A technique for efficiently propagating beneficial changes across all system modules quickly
B. A design pattern that ensures changes in one module are automatically applied everywhere now
C. Changes in one part of the system causing unintended effects in seemingly unrelated parts
D. A method for testing software by sending signals through all connected components sequentially

Answer: C

Q747. What is program comprehension and why is it critical for maintenance?

A. Understanding existing code is essential for making correct modifications without introducing bugs
B. Program comprehension is unnecessary because maintainers can change code without understanding it
C. Program comprehension only matters for documentation purposes and not for actual code changes
D. Understanding code is only important during initial development and never during maintenance work

Answer: A

Q748. What role does impact analysis play in software maintenance?

A. It identifies all system areas affected by a proposed change before implementation begins now
B. It only identifies positive impacts and ignores any negative consequences of proposed changes
C. It is only performed after changes are deployed to assess the damage caused by the change
D. It has no role in maintenance because changes can be made without considering their effects

Answer: A

Q749. How does automated testing support software maintenance activities?

A. It eliminates the need for any human involvement in the maintenance decision-making process
B. It replaces all maintenance activities including code modification and deployment processes
C. It quickly validates that changes have not broken existing functionality through regression tests
D. It automatically fixes all bugs found during testing without any developer intervention needed

Answer: C

Q750. What is software rejuvenation?

A. Adding cosmetic visual updates to the user interface without any functional changes at all
B. Permanently retiring software and replacing it with an entirely new system from scratch now
C. Periodically restarting or refreshing software to counteract performance degradation over time
D. Increasing the hardware resources allocated to the software without changing any code at all

Answer: C

Q751. What is the CMMI (Capability Maturity Model Integration)?

A. A hardware testing tool for measuring processor speed and memory performance now
B. A database management system used exclusively for storing quality audit records
C. A specific programming language designed for building quality management applications
D. A framework for assessing and improving organizational process maturity and capability

Answer: D

Q752. What are the five maturity levels in CMMI?

A. Requirements, Design, Implementation, Testing, and Deployment of system products
B. Planning, Coding, Testing, Deploying, and Maintaining software applications now
C. Beginning, Intermediate, Advanced, Expert, and Master skill certification levels
D. Initial, Managed, Defined, Quantitatively Managed, and Optimizing across all processes

Answer: D

Q753. How does statistical process control apply to software quality assurance?

A. It has no application to software because software processes cannot be measured now
B. It only applies to manufacturing processes and is irrelevant to software development
C. It uses statistical methods to monitor and control process performance and variation
D. It replaces all testing activities with purely statistical models of defect prediction

Answer: C

Q754. What is the cost of quality (CoQ) model?

A. A pricing strategy for selling software products to customers at premium prices now
B. A budget allocation method assigning equal funding to all project activities always
C. A salary structure for paying quality assurance engineers based on defects found
D. A model categorizing quality costs into prevention, appraisal, and failure categories

Answer: D

Q755. How do formal technical reviews improve software quality?

A. They replace all automated testing by manually verifying every line of source code now
B. They systematically examine work products to find defects before they propagate further
C. They only focus on coding style and have no impact on finding functional defect issues
D. They are informal conversations without structure that rarely produce useful findings

Answer: B

Q756. What is the relationship between process quality and product quality?

A. Poor processes always produce high-quality products because developers compensate fully
B. Good processes increase the likelihood of producing good products consistently over time
C. Process quality has no correlation with product quality in software engineering at all
D. Product quality depends solely on individual developer skill and not on process at all

Answer: B

Q757. What is a quality gate in the development process?

A. A software tool that automatically generates test cases from requirements documents
B. A checkpoint where quality criteria must be met before proceeding to the next phase
C. A physical entrance to the quality assurance department office in the building now
D. A coding standard that restricts the number of lines per function in source code

Answer: B

Q758. How does root cause analysis improve quality assurance?

A. It identifies underlying causes of defects to prevent similar issues from recurring again
B. It has no practical application in software quality assurance or defect prevention work
C. It replaces all testing activities by predicting where defects will occur in source code
D. It only identifies surface-level symptoms without investigating the deeper reasons for them

Answer: A

Q759. What is the difference between verification and validation in QA?

A. Verification only applies to code; validation only applies to documentation artifacts
B. Verification ensures correct construction; validation ensures the right product is built
C. Verification and validation are identical activities performed at the same project stage
D. Verification is done by users; validation is done by developers during the development

Answer: B

Q760. What is a software quality assurance plan (SQAP)?

A. A document defining quality activities, standards, and procedures for a project effort
B. A hardware procurement list for purchasing testing equipment and tools for the lab
C. A project schedule showing only the testing phase timeline without any other phases
D. A source code file containing automated tests for the application being developed now

Answer: A

Q761. How is function point analysis different from LOC as a size metric?

A. Function points measure functionality delivered; LOC measures physical source code volume size
B. LOC is technology-independent while function points depend on the programming language used
C. Function points are less accurate than LOC because they ignore the actual code implementation
D. Function points and LOC always produce identical size measurements for any given software system

Answer: A

Q762. What is the Halstead complexity metric based on?

A. The number of operators and operands in the source code of the program being measured today
B. The number of developers who have contributed to the source code repository over project life
C. The amount of time required to execute the software on a standard hardware configuration now
D. The physical size of the compiled executable file measured in megabytes of storage space used

Answer: A

Q763. What is the difference between leading and lagging metrics?

A. Lagging metrics predict future outcomes; leading metrics only describe what happened previously
B. Leading and lagging metrics are identical concepts measuring the same things at the same time
C. Leading metrics predict future outcomes; lagging metrics report on past results already achieved
D. Leading metrics are less useful than lagging metrics because predictions are always inaccurate

Answer: C

Q764. How does the Goal-Question-Metric (GQM) approach structure measurement programs?

A. It starts by collecting all possible metrics and then determines if any goals can be derived
B. It defines goals first, then questions to assess goals, then metrics to answer those questions
C. It requires implementing all metrics simultaneously before defining any organizational goals
D. It randomly selects metrics without any connection to organizational goals or questions asked

Answer: B

Q765. What is coupling metrics and how does it relate to maintainability?

A. Coupling can only be measured for object-oriented systems and not for any other paradigms
B. Higher coupling always improves maintainability because modules share more common knowledge
C. Coupling metrics have no relationship with software maintainability or quality in any context
D. Coupling metrics measure interdependence between modules; higher coupling reduces maintainability

Answer: D

Q766. What is the purpose of software reliability metrics?

A. To track the number of meetings held by the development team during the project timeline
B. To measure the visual attractiveness of the user interface design and layout of the system
C. To count the number of programming languages used in the software project source code now
D. To quantify the probability of failure-free operation over a specified time period and context

Answer: D

Q767. How does maintainability index combine multiple metrics?

A. It combines cyclomatic complexity, LOC, and Halstead volume into a single composite score value
B. It measures only the number of comments in code as the sole indicator of maintainability level
C. It only uses lines of code without considering any other metrics for maintainability assessment
D. It counts the number of developers who can understand the code without any other measurements

Answer: A

Q768. What is the difference between objective and subjective software metrics?

A. Subjective metrics can be fully automated while objective metrics always require human judgment
B. Objective and subjective metrics are identical and always produce the same measurement results
C. Objective metrics are less accurate than subjective metrics in all measurement scenarios today
D. Objective metrics are directly measurable; subjective metrics require human judgment and opinion

Answer: D

Q769. What is technical debt measurement and how can it be quantified?

A. Technical debt cannot be measured or quantified in any meaningful way using any known methods
B. Technical debt measurement counts the number of developers who have left the project team
C. It only measures the financial debt of the company and has no relation to software quality now
D. It estimates the cost of fixing shortcuts and workarounds accumulated in the codebase over time

Answer: D

Q770. How do dashboard metrics help project managers make decisions?

A. They automatically make all project decisions without requiring any human judgment or input
B. They only display historical data and cannot help with current or future decision making work
C. They replace the need for any communication between project managers and development teams
D. They provide real-time visibility into project health enabling data-driven decision making today

Answer: D

Q771. What is the difference between centralized and distributed version control?

A. Centralized systems are always superior to distributed systems for all development scenarios now
B. Distributed systems require a constant network connection while centralized systems do not need
C. Centralized uses one server repository; distributed gives each developer a full local repository
D. Centralized and distributed version control systems are identical in architecture and capability

Answer: C

Q772. What is a merge conflict and how is it resolved?

A. Merge conflicts only happen in centralized systems and never in distributed version control
B. It occurs when changes overlap and requires manual resolution by choosing the correct changes
C. Merge conflicts never occur in modern version control systems due to advanced automation today
D. Conflicts are always resolved automatically without any human intervention being required now

Answer: B

Q773. What is continuous delivery (CD) in configuration management?

A. Manually deploying software to production once every year during the scheduled maintenance window
B. Automating the release process so software can be deployed to production at any time reliably
C. Delivering hardware components continuously to the data center for server infrastructure setup
D. Preventing any changes from reaching production until all features are completely implemented

Answer: B

Q774. What is the purpose of tagging in version control?

A. Labeling hardware components in the server rack for identification during maintenance periods
B. Assigning developers to specific tasks using labels in the project management tracking system
C. Adding metadata to database records for improved search and retrieval performance in queries
D. Marking specific points in the repository history as important like releases or milestones now

Answer: D

Q775. How does GitFlow branching strategy organize development work?

A. It uses feature, develop, release, and hotfix branches with defined merge workflows for teams
B. It creates a new repository for each feature instead of using branches within one repository
C. It uses a single branch for all development without any branching or merging strategies now
D. It prohibits branching entirely and requires all developers to commit directly to the main line

Answer: A

Q776. What is the role of build automation in configuration management?

A. It eliminates the need for any testing by guaranteeing that automated builds are always correct
B. It only manages the visual design of the user interface without any involvement in code builds
C. It ensures consistent, repeatable builds by automating compilation, testing, and packaging steps
D. It replaces version control by automatically generating source code from requirements documents

Answer: C

Q777. What is a configuration audit?

A. A performance test measuring the application response time under peak load conditions only now
B. A code review focused exclusively on the visual appearance of the user interface elements now
C. A financial audit of the project budget to ensure money is being spent according to the plan
D. A review verifying that configuration items conform to their documented specifications and records

Answer: D

Q778. What is trunk-based development?

A. All development occurs in release branches without any central trunk or main branch at all now
B. Developers never commit code and instead share changes through email attachments exclusively
C. All developers commit to a single main branch with short-lived feature branches if needed now
D. Each developer maintains their own permanent long-lived branch that is never merged with others

Answer: C

Q779. How does semantic versioning (SemVer) communicate change significance?

A. All version numbers are randomly assigned without any meaning related to the type of changes
B. Only the major version number is meaningful while minor and patch numbers are decorative only
C. Semantic versioning uses alphabetical labels instead of numbers to indicate version differences
D. Major version for breaking changes, minor for features, and patch for bug fixes are indicated

Answer: D

Q780. What is the purpose of a deployment pipeline?

A. To design the user interface mockups that will be implemented by the development team later
B. To generate project management reports for stakeholder review at monthly status update meetings
C. To manually review every line of code before it is committed to the version control repository
D. To automate and validate the stages between code commit and production deployment reliably

Answer: D

Q781. How is a risk exposure value calculated?

A. By adding the total project budget to the number of team members assigned
B. By multiplying the probability of occurrence by the estimated impact value
C. By subtracting the actual cost from the planned budget at each milestone
D. By dividing the number of defects found by the total lines of code written

Answer: B

Q782. What is the difference between risk transfer and risk avoidance?

A. Transfer shifts risk to another party; avoidance eliminates the risk from the project entirely
B. Transfer and avoidance are identical strategies with no differences in their approaches at all
C. Neither transfer nor avoidance has any effect on the actual risk profile of the project work
D. Transfer eliminates risks completely; avoidance shifts them to another responsible party now

Answer: A

Q783. What is a risk-driven development approach?

A. Prioritizing development activities based on the highest identified risk areas in the project
B. Using risk management only during the final testing phase and not during development work
C. Ignoring all risks and developing features in random order without any priority criteria now
D. Developing only low-risk features and permanently deferring all high-risk items from scope

Answer: A

Q784. How does the Boehm risk management model categorize project risks?

A. Into top ten risk items that are continuously tracked and mitigated throughout the project
B. Into risks that only affect the schedule and risks that only affect the budget separately
C. Into a single list of risks that is created once and never updated during the project life
D. Into exactly two categories of risks that are either accepted or completely eliminated now

Answer: A

Q785. What is sensitivity analysis in risk management?

A. Measuring the emotional response of team members to project deadlines and pressure levels
B. Determining which risks have the greatest potential impact on project outcomes and results
C. Testing the application's response time under varying network latency and bandwidth levels
D. Evaluating the sensitivity of hardware components to environmental conditions in data center

Answer: B

Q786. What is the purpose of risk monitoring and control?

A. Creating the initial risk register at the beginning of the project and never updating it again
B. Focusing exclusively on positive risks and ignoring all negative risk events in the project
C. Tracking identified risks, detecting new risks, and evaluating risk response effectiveness now
D. Eliminating all monitoring activities to reduce overhead and speed up project delivery today

Answer: C

Q787. How do qualitative and quantitative risk analysis differ?

A. Quantitative analysis does not use any numerical values and relies solely on expert opinion
B. Qualitative and quantitative analysis produce identical results using the same methods always
C. Qualitative assesses risks subjectively by category; quantitative assigns numerical probabilities
D. Qualitative analysis is always more accurate than quantitative analysis in every situation now

Answer: C

Q788. What is a risk response strategy?

A. A technique for ignoring risks to maintain team morale and project momentum during development
B. A financial strategy for investing project savings into high-risk stock market opportunities
C. A planned approach for addressing identified risks through avoidance, mitigation, or transfer
D. A reactive action taken only after a risk has already materialized and caused project damage

Answer: C

Q789. What is the role of risk communication in project management?

A. Ensuring all stakeholders are informed about risks and their management throughout the project
B. Delegating all risk communication to the most junior team member without any oversight given
C. Communicating only positive project news and avoiding any mention of potential risk events now
D. Hiding risk information from stakeholders to prevent unnecessary worry about project outcomes

Answer: A

Q790. What is a risk breakdown structure (RBS)?

A. A technique for splitting the project budget into smaller allocations for each team member
B. A method for breaking down source code into smaller functions for better risk assessment work
C. A physical diagram showing the location of server hardware in the data center for risk review
D. A hierarchical categorization of project risks organized by source and type for clarity now

Answer: D

Q791. What is SQL injection and how can it be prevented?

A. An approved method for administrators to directly modify production database records quickly
B. A database optimization technique that improves query performance through code injection method
C. A testing technique where SQL queries are injected into the test suite for validation work
D. Inserting malicious SQL through user inputs; prevented by parameterized queries and validation

Answer: D

Q792. What is cross-site scripting (XSS) vulnerability?

A. A method for testing website compatibility across different web browsers and platforms now
B. A programming pattern for organizing JavaScript code across multiple source code files today
C. Injecting malicious scripts into web pages that execute in other users' browsers unexpectedly
D. A legitimate technique for sharing scripts between different websites for code reuse purposes

Answer: C

Q793. What is the OWASP Top Ten?

A. A ranking of the ten best programming languages for building secure web applications now
B. A certification program for security engineers with ten levels of professional expertise
C. A list of the most critical web application security risks updated periodically by experts
D. A collection of ten design patterns used exclusively for building user interface components

Answer: C

Q794. What is defense in depth as a security strategy?

A. Removing all security controls to simplify the application architecture and reduce costs
B. Implementing security only at the network perimeter without any application-level controls
C. Relying on a single strong security control to protect all application assets and data now
D. Using multiple layers of security controls so that if one layer fails others still protect

Answer: D

Q795. What is the purpose of threat modeling?

A. Creating mathematical models of the project schedule to predict the delivery date precisely
B. Designing the database schema using entity-relationship diagrams for data storage planning
C. Modeling the visual appearance of the application user interface in three-dimensional space
D. Systematically identifying potential threats and vulnerabilities in the system architecture now

Answer: D

Q796. What is secure coding practice?

A. Writing code as fast as possible without considering any security implications or standards
B. Writing code that follows security guidelines to prevent common vulnerabilities and exploits
C. Avoiding all input validation to improve the user experience and reduce form friction now
D. Using only deprecated libraries because they have been tested longer than newer versions

Answer: B

Q797. What is the role of penetration testing in security engineering?

A. Replacing all other security measures because penetration testing finds every possible issue
B. Evaluating the visual design of the user interface for compliance with branding guidelines
C. Simulating real-world attacks to identify exploitable vulnerabilities before attackers find them
D. Testing the application performance under heavy network traffic load conditions and stress

Answer: C

Q798. What is a security token used for?

A. Storing application source code in an encrypted format within the version control system
B. Designing the database schema with proper normalization and referential integrity rules
C. Providing a secure method for authenticating and authorizing user sessions and API access
D. Measuring the performance of the application under concurrent user load conditions now

Answer: C

Q799. What is the difference between symmetric and asymmetric encryption?

A. There is no difference between symmetric and asymmetric encryption in practice or theory now
B. Symmetric encryption is always stronger than asymmetric encryption in every possible scenario
C. Asymmetric encryption uses one shared key; symmetric encryption uses a public-private key pair
D. Symmetric uses one shared key for both operations; asymmetric uses a public and private key pair

Answer: D

Q800. What is input validation and why is it critical for security?

A. Input validation is a visual design concern that has no relationship to application security
B. Input validation slows down the application and should be avoided for performance reasons now
C. Checking user inputs against expected formats to prevent injection attacks and data corruption
D. It only applies to database inputs and is not needed for any other type of user interaction

Answer: C

Q801. What is the key difference between software engineering and system engineering?

A. System engineering only deals with hardware
B. Software engineering focuses on software; system engineering covers the entire socio-technical system
C. They are the same discipline
D. System engineering is a subset of software engineering

Answer: B

Q802. What is the purpose of a software engineering code of ethics?

A. To enforce programming standards
B. To guide professional behavior and responsibility toward society
C. To define coding syntax rules
D. To set salary standards for engineers

Answer: B

Q803. Which type of software application involves processing large volumes of business data?

A. Embedded software
B. System software
C. Business application software
D. Real-time software

Answer: C

Q804. What does software evolution refer to?

A. Rewriting software from scratch
B. The process of changing software after delivery to correct faults or improve performance
C. Installing new software
D. The initial development phase

Answer: B

Q805. Which challenge is specific to web-based software engineering?

A. Lack of programming languages
B. Need for scalability, security, and rapid deployment
C. Inability to use databases
D. No need for user interfaces

Answer: B

Q806. What is the significance of the NATO Software Engineering Conference of 1968?

A. It introduced the Java programming language
B. It formally recognized software engineering as a discipline
C. It created the internet
D. It established the first software company

Answer: B

Q807. What is meant by software reuse?

A. Recycling old hardware
B. Using existing software components to build new systems
C. Copying code without permission
D. Reinstalling software on a different machine

Answer: B

Q808. What is emergent system property?

A. A property visible in source code
B. A property that arises from the interaction of system components and cannot be attributed to any single component
C. A bug that appears randomly
D. A feature added after release

Answer: B

Q809. Which factor has the MOST influence on software project failure?

A. Using the wrong programming language
B. Poor requirements management
C. Using too many monitors
D. Team members being in the same office

Answer: B

Q810. What is the relationship between software engineering and computer science?

A. They are identical fields
B. Computer science provides theoretical foundations; software engineering applies them practically
C. Computer science is a branch of software engineering
D. They have no relationship

Answer: B

Q811. How does the V-model differ from the traditional Waterfall model?

A. The V-model has no testing phase
B. The V-model explicitly associates each development phase with a corresponding testing phase
C. The V-model eliminates design
D. They are exactly the same

Answer: B

Q812. What is the primary risk addressed by the Spiral model?

A. Lack of programmers
B. Project risks identified through repeated risk analysis in each spiral iteration
C. Poor internet connectivity
D. Office space constraints

Answer: B

Q813. What is evolutionary prototyping?

A. A prototype that is immediately discarded
B. A prototype that is incrementally refined and evolves into the final system
C. A testing technique
D. A documentation method

Answer: B

Q814. In RAD (Rapid Application Development), what enables faster development?

A. Skipping all testing
B. Using component-based construction and automated tools with intensive user involvement
C. Writing code without design
D. Eliminating documentation entirely

Answer: B

Q815. What is the key disadvantage of the Waterfall model?

A. It is too fast
B. Difficulty accommodating changes after a phase is complete
C. It requires no documentation
D. It has too many iterations

Answer: B

Q816. What is a timeboxed iteration?

A. An iteration with no deadline
B. An iteration with a fixed, predetermined duration that cannot be extended
C. An iteration that lasts forever
D. A phase that only runs at night

Answer: B

Q817. How does the incremental model handle changing requirements?

A. It cannot handle changes
B. Later increments can incorporate new or changed requirements without affecting delivered increments
C. It restarts from scratch
D. It ignores changes completely

Answer: B

Q818. What is a process framework in software engineering?

A. A programming library
B. A foundation that defines the activities, actions, and tasks for a complete software process
C. A testing tool
D. A type of database

Answer: B

Q819. What is the primary advantage of using the Spiral model?

A. It is the simplest model
B. It combines iterative development with systematic risk management
C. It never requires customer involvement
D. It eliminates all project risks

Answer: B

Q820. What distinguishes a prescriptive process model from an adaptive one?

A. Prescriptive models have no defined steps
B. Prescriptive models define a specific set of activities in a defined order; adaptive models allow flexibility
C. Adaptive models never deliver software
D. There is no difference

Answer: B

Q821. What is the concept of 'velocity' in Agile?

A. How fast the team types
B. The amount of work a team can complete in one iteration, measured in story points or similar units
C. The speed of the internet connection
D. The number of meetings per week

Answer: B

Q822. What is pair programming in Extreme Programming (XP)?

A. Two teams working on separate projects
B. Two developers working together at one workstation, one writing code and the other reviewing in real time
C. Two developers in different countries
D. A code review done after deployment

Answer: B

Q823. What is the purpose of Sprint Planning in Scrum?

A. To assign blame for past failures
B. To select items from the Product Backlog and define the work plan for the upcoming Sprint
C. To write the final project report
D. To conduct user acceptance testing

Answer: B

Q824. What is a Kanban board used for in Agile?

A. Drawing architectural diagrams
B. Visualizing work in progress by showing tasks moving through workflow stages
C. Tracking employee vacation days
D. Managing email

Answer: B

Q825. What is continuous refactoring in Agile?

A. Rewriting the entire system regularly
B. Ongoing improvement of code structure and design without changing its external behavior
C. Continuously adding new features
D. Removing features from the product

Answer: B

Q826. What is the role of the Scrum Master during a Sprint?

A. Assigning tasks to developers
B. Facilitating the Scrum process, removing impediments, and shielding the team from external distractions
C. Writing code exclusively
D. Approving the final product

Answer: B

Q827. What is an epic in Agile?

A. A very short user story
B. A large body of work that can be broken down into smaller user stories
C. A type of software bug
D. A deployment milestone

Answer: B

Q828. What is the purpose of a Sprint Backlog?

A. To list all company projects
B. To contain the selected Product Backlog items and the plan for delivering them during the Sprint
C. To track customer complaints
D. To record team vacation schedules

Answer: B

Q829. What is the MoSCoW prioritization technique?

A. A Russian programming method
B. A method categorizing requirements as Must have, Should have, Could have, and Won't have
C. A testing framework
D. A deployment strategy

Answer: B

Q830. What is a use case scenario?

A. A test case for unit testing
B. A specific sequence of interactions between an actor and the system to achieve a particular goal
C. A deployment plan
D. A database schema

Answer: B

Q831. What is the purpose of a requirements traceability matrix?

A. To trace network packets
B. To map each requirement to its source, design elements, and test cases
C. To track employee attendance
D. To monitor server uptime

Answer: B

Q832. What is requirements ambiguity?

A. Requirements that are too detailed
B. Requirements that can be interpreted in more than one way
C. Requirements that are too short
D. Requirements written in code

Answer: B

Q833. What is the difference between data flow diagrams (DFDs) and use case diagrams?

A. They are identical
B. DFDs show how data moves through the system; use case diagrams show interactions between users and system functions
C. DFDs are only for databases
D. Use case diagrams show data flows

Answer: B

Q834. What is a JAD (Joint Application Development) session?

A. A solo programming activity
B. A facilitated workshop bringing stakeholders and developers together to define requirements collaboratively
C. A hardware testing procedure
D. A deployment meeting

Answer: B

Q835. What is the role of a domain model in requirements engineering?

A. To model database tables
B. To represent key concepts, relationships, and rules of the problem domain to facilitate shared understanding
C. To model network topology
D. To design user interfaces

Answer: B

Q836. What is the difference between structured and unstructured requirements interviews?

A. There is no difference
B. Structured interviews follow a predefined set of questions; unstructured interviews allow open-ended exploration
C. Structured interviews have no questions prepared
D. Unstructured interviews use questionnaires

Answer: B

Q837. What is the difference between top-down and bottom-up estimation?

A. They produce identical results
B. Top-down estimates the whole project then breaks it down; bottom-up estimates individual tasks then aggregates them
C. Top-down is always more accurate
D. Bottom-up ignores individual tasks

Answer: B

Q838. What is Brooks' Law?

A. Adding more programmers always helps
B. Adding manpower to a late software project makes it later due to communication overhead and ramp-up time
C. Projects always finish on time
D. Larger teams produce fewer bugs

Answer: B

Q839. What is the purpose of a Work Breakdown Structure (WBS)?

A. To break down the team hierarchy
B. To decompose the project into smaller, manageable work packages that can be estimated and tracked
C. To break software into functions
D. To break hardware into components

Answer: B

Q840. What is analogous estimation?

A. Estimating by writing code first
B. Using data from similar past projects to estimate the current project's effort and cost
C. Random guessing
D. Estimating based on lines of code only

Answer: B

Q841. What is project tracking and control?

A. Tracking employee locations
B. Monitoring project progress against the plan and taking corrective action when deviations occur
C. Controlling access to the server room
D. Tracking software downloads

Answer: B

Q842. What is the difference between a Gantt chart and a PERT chart?

A. They are identical
B. A Gantt chart shows tasks as horizontal bars on a timeline; a PERT chart shows task dependencies as a network diagram
C. Gantt charts show only costs
D. PERT charts cannot show time

Answer: B

Q843. What is the concept of 'slack time' in project scheduling?

A. Time spent not working
B. The amount of time an activity can be delayed without delaying the overall project completion date
C. Time reserved for meetings
D. Break time for developers

Answer: B

Q844. What is the difference between composition and inheritance in OO design?

A. They are the same concept
B. Inheritance creates an 'is-a' relationship; composition creates a 'has-a' relationship by containing objects
C. Composition is always worse
D. Inheritance is never used

Answer: B

Q845. What is the Template Method design pattern?

A. A template for writing documentation
B. A pattern that defines the skeleton of an algorithm in a superclass, deferring specific steps to subclasses
C. A method for creating templates
D. A UI design template

Answer: B

Q846. What is the difference between functional and object-oriented decomposition?

A. They are identical approaches
B. Functional decomposition breaks the system into functions; OO decomposition breaks it into objects that encapsulate data and behavior
C. Functional decomposition is always better
D. OO decomposition ignores data

Answer: B

Q847. What is the State design pattern?

A. A pattern for managing US states
B. A pattern that allows an object to alter its behavior when its internal state changes, appearing to change its class
C. A pattern for database state management
D. A pattern for version control

Answer: B

Q848. What is the difference between data-centered and data-flow architecture in design?

A. They are the same
B. Data-centered architecture has a central repository accessed by components; data-flow architecture processes data through a sequence of transformations
C. Data-centered ignores data
D. Data-flow never processes data

Answer: B

Q849. What is the Proxy design pattern?

A. A network proxy server
B. A pattern that provides a surrogate or placeholder for another object to control access to it
C. A debugging tool
D. A testing framework

Answer: B

Q850. What is the difference between horizontal and vertical scaling?

A. They are the same
B. Horizontal scaling adds more machines; vertical scaling adds more power (CPU, RAM) to existing machines
C. Horizontal scaling reduces capacity
D. Vertical scaling always requires new hardware

Answer: B

Q851. What is a message queue in distributed architecture?

A. A queue for email messages
B. An asynchronous communication mechanism that stores messages until the receiving component is ready to process them
C. A task assignment queue
D. A debugging tool

Answer: B

Q852. What is the Model-View-Controller (MVC) pattern?

A. A version control model
B. An architectural pattern that separates an application into Model (data), View (UI), and Controller (input handling)
C. A testing methodology
D. A project management framework

Answer: B

Q853. What is an API gateway in microservices architecture?

A. A physical network gateway
B. A single entry point that routes client requests to appropriate microservices and handles cross-cutting concerns
C. A database connector
D. A code compiler

Answer: B

Q854. What is the difference between SOA and microservices architecture?

A. They are identical
B. SOA uses an enterprise service bus and shares data models; microservices use lightweight protocols and each service owns its data
C. SOA is always better
D. Microservices never use APIs

Answer: B

Q855. What is a load balancer in software architecture?

A. A device that weighs servers
B. A component that distributes incoming network traffic across multiple servers to ensure no single server is overwhelmed
C. A code optimization tool
D. A testing utility

Answer: B

Q856. What is a service mesh in microservices?

A. A physical network mesh
B. A dedicated infrastructure layer that handles service-to-service communication, including load balancing, encryption, and observability
C. A type of database
D. A code generation tool

Answer: B

Q857. What is a containerized architecture?

A. An architecture stored in shipping containers
B. An approach where applications are packaged with their dependencies in lightweight containers for consistent deployment across environments
C. An architecture using physical boxes
D. A hardware architecture

Answer: B

Q858. What is the difference between a persona and a user profile?

A. They are the same thing
B. A persona is a fictional representative character based on research; a user profile is actual data about a specific user
C. Personas are always inaccurate
D. User profiles are fictional

Answer: B

Q859. What is the principle of consistency in UI design?

A. Changing the UI frequently
B. Using the same design patterns, terminology, and interactions throughout the interface so users can predict behavior
C. Making each page look different
D. Randomizing button positions

Answer: B

Q860. What is an A/B test in UI design?

A. A test with only two answers
B. A method of comparing two versions of a UI element to determine which performs better based on user metrics
C. A binary code test
D. An accessibility test

Answer: B

Q861. What is a card sorting exercise used for?

A. Sorting playing cards
B. A user research method where participants organize content into categories to inform information architecture
C. Sorting database records
D. Organizing code files

Answer: B

Q862. What is the difference between accessibility and usability?

A. They are identical
B. Accessibility ensures the interface is usable by people with disabilities; usability measures how easy and efficient the interface is for all users
C. Accessibility is optional
D. Usability is only about speed

Answer: B

Q863. What is the purpose of breadcrumb navigation?

A. Tracking user cookies
B. Showing users their current location within the site hierarchy and allowing easy navigation to parent pages
C. Displaying error logs
D. Managing user sessions

Answer: B

Q864. What is progressive disclosure in UI design?

A. Gradually revealing source code
B. Showing only essential information initially and revealing additional details as the user needs them
C. Disclosing security vulnerabilities
D. A type of animation effect

Answer: B

Q865. What are Nielsen's error prevention heuristics about?

A. Preventing hardware errors
B. Designing interfaces that prevent errors from occurring in the first place, such as confirmation dialogs and input validation
C. Preventing network outages
D. Preventing code compilation errors

Answer: B

Q866. What is code coverage and why is it important?

A. The number of files in a project
B. A metric measuring the percentage of code executed during testing, indicating how thoroughly the code is tested
C. How many developers work on the code
D. The number of code comments

Answer: B

Q867. What is the difference between static and dynamic code analysis?

A. They are identical
B. Static analysis examines code without executing it; dynamic analysis examines code behavior during execution
C. Static analysis is always slower
D. Dynamic analysis does not require running the code

Answer: B

Q868. What is dependency injection?

A. Injecting malicious code
B. A technique where an object receives its dependencies from external sources rather than creating them internally
C. Adding new library imports
D. A database insertion technique

Answer: B

Q869. What is a code smell?

A. Physical odor from a computer
B. A surface indication in code that suggests a deeper design problem requiring refactoring
C. A compilation error
D. A runtime crash

Answer: B

Q870. What is a linter in software development?

A. A piece of clothing fabric
B. A tool that analyzes source code to flag programming errors, bugs, stylistic issues, and suspicious constructs
C. A compiler optimization
D. A deployment tool

Answer: B

Q871. What is the YAGNI principle?

A. You Are Going to Need It
B. You Aren't Gonna Need It: do not implement functionality until it is actually needed
C. Yet Another Great New Innovation
D. Your Application Gets New Issues

Answer: B

Q872. What is technical debt?

A. Money owed for software licenses
B. The implied cost of future rework caused by choosing a quick solution now instead of a better approach that would take longer
C. The cost of hardware
D. Unpaid developer salaries

Answer: B

Q873. What is a code review and why is it important?

A. Reviewing the project budget
B. A systematic examination of source code by peers to identify defects, improve quality, and share knowledge
C. A customer feedback session
D. A marketing review

Answer: B

Q874. What is the difference between top-down and bottom-up integration testing?

A. They are identical approaches
B. Top-down tests from the main module downward using stubs; bottom-up tests from the lowest modules upward using drivers
C. Top-down is always better
D. Bottom-up never requires drivers

Answer: B

Q875. What is statement coverage in testing?

A. Coverage of requirement statements
B. A metric measuring the percentage of executable code statements that have been exercised by test cases
C. The number of comment lines tested
D. Coverage of variable declarations

Answer: B

Q876. What is branch coverage and how does it differ from statement coverage?

A. They are the same metric
B. Branch coverage ensures every branch of every decision point is executed, providing stronger coverage than statement coverage alone
C. Branch coverage is weaker than statement coverage
D. Branch coverage only applies to switch statements

Answer: B

Q877. What is the purpose of a test harness?

A. A horse harness for testing
B. A framework that provides the infrastructure for automating test execution, including drivers, stubs, and result reporting
C. A physical testing device
D. A documentation tool

Answer: B

Q878. What is the difference between smoke testing and sanity testing?

A. They are the same thing
B. Smoke testing verifies that critical functions work after a build; sanity testing verifies specific functionality after a minor change
C. Smoke testing uses fire
D. Sanity testing tests mental health

Answer: B

Q879. What is the purpose of a test stub?

A. A ticket stub for testing events
B. A simplified implementation of a component that a module under test depends on, providing predefined responses
C. A type of test case
D. A testing report

Answer: B

Q880. What is the concept of 'test automation'?

A. Robots performing manual testing
B. Using software tools and scripts to execute tests automatically, compare results, and report outcomes
C. Automating the hiring of testers
D. Automatically generating code

Answer: B

Q881. What is the difference between load testing and stress testing?

A. They are identical
B. Load testing measures performance under expected conditions; stress testing pushes the system beyond normal capacity to find its breaking point
C. Load testing is easier
D. Stress testing is always done first

Answer: B

Q882. What is impact analysis in software maintenance?

A. Analyzing the impact of a meteor
B. Evaluating the consequences of a proposed change on the rest of the system to identify affected components and potential risks
C. Analyzing market impact
D. Analyzing team morale

Answer: B

Q883. What is program comprehension in maintenance?

A. Understanding a TV program
B. The process by which maintainers understand the existing code's structure, behavior, and intent before making modifications
C. A reading comprehension test
D. Understanding user requirements

Answer: B

Q884. What is software migration?

A. Software moving to another country
B. The process of transferring a software system from one operating environment to another while preserving functionality
C. Developers changing jobs
D. Moving files between folders

Answer: B

Q885. What is the concept of 'maintainability index'?

A. A measure of building maintenance costs
B. A composite metric that combines code complexity, volume, and other factors to quantify how easy the software is to maintain
C. An employee performance metric
D. A hardware reliability metric

Answer: B

Q886. What is the role of regression testing in maintenance?

A. It has no role
B. Ensuring that maintenance changes have not introduced new defects into previously working functionality
C. Testing only new features
D. Testing only the user interface

Answer: B

Q887. What is code restructuring?

A. Moving code files to different folders
B. Modifying the internal structure of existing code to improve readability and maintainability without changing its external behavior
C. Rewriting code in a different language
D. Deleting unused code

Answer: B

Q888. What is the challenge of maintaining undocumented code?

A. There is no challenge
B. Maintainers must spend significant time understanding the code's purpose and logic through code reading alone, increasing the risk of introducing errors
C. Undocumented code runs faster
D. Undocumented code is easier to maintain

Answer: B

Q889. What is the difference between emergency and scheduled maintenance?

A. They are the same
B. Emergency maintenance addresses critical failures requiring immediate fixes; scheduled maintenance is planned changes during designated windows
C. Emergency maintenance is optional
D. Scheduled maintenance is never planned

Answer: B

Q890. What is a Service Level Agreement (SLA) in the context of maintenance?

A. A software license agreement
B. A contract defining expected maintenance response times, resolution times, and availability targets
C. A developer employment contract
D. A hardware warranty

Answer: B

Q891. What is a formal technical review (FTR)?

A. A review of technical documents by lawyers
B. A structured review meeting where a team of technical staff evaluates a work product to identify defects and verify quality
C. A hardware inspection
D. A user acceptance review

Answer: B

Q892. What is defect density as a quality metric?

A. The physical density of a computer
B. The number of confirmed defects found per unit of software size, such as defects per thousand lines of code
C. The number of defects per developer
D. The weight of defect reports

Answer: B

Q893. What is the Six Sigma approach in software quality?

A. A Greek letter used in equations
B. A data-driven methodology aiming to reduce defects to fewer than 3.4 per million opportunities using DMAIC
C. A programming framework
D. A type of testing

Answer: B

Q894. What is root cause analysis in QA?

A. Analyzing tree roots
B. A systematic process of identifying the fundamental reason why a defect occurred to prevent similar defects in the future
C. Analyzing code roots
D. A type of testing

Answer: B

Q895. What is the purpose of quality gates in a development process?

A. Physical gates at the office entrance
B. Checkpoints where work products must meet predefined quality criteria before proceeding to the next phase
C. Gates in logic circuits
D. Entrance exams for QA engineers

Answer: B

Q896. What is the Pareto analysis used for in QA?

A. Analyzing Italian software
B. Identifying the vital few causes that account for the majority of defects, focusing improvement efforts where they have the most impact
C. Analyzing parrot behavior
D. A type of performance testing

Answer: B

Q897. What is the ISO 9001 standard in relation to software quality?

A. A programming language standard
B. An international standard for quality management systems that defines requirements for consistent product quality and customer satisfaction
C. A database standard
D. A network protocol standard

Answer: B

Q898. What is the role of metrics in quality assurance?

A. Metrics have no role in QA
B. Metrics provide objective, quantitative data to measure process effectiveness, product quality, and guide improvement decisions
C. Metrics only measure project cost
D. Metrics replace testing

Answer: B

Q899. How is cyclomatic complexity calculated?

A. By counting lines of code
B. By counting the number of linearly independent paths through the code, calculated as V(G) = E - N + 2P for the control flow graph
C. By counting the number of classes
D. By counting the number of variables

Answer: B

Q900. What does a high cyclomatic complexity value indicate?

A. Simple, easy-to-test code
B. Code with many decision paths that is more complex, harder to test, and more error-prone
C. Fast-running code
D. Well-documented code

Answer: B

Q901. What is function point analysis used for?

A. Analyzing mathematical functions
B. Measuring software size based on the functionality delivered to the user, independent of the programming language
C. Analyzing function calls in code
D. Measuring hardware performance

Answer: B

Q902. What are the five components measured in function point analysis?

A. Classes, methods, variables, loops, conditions
B. External Inputs, External Outputs, External Inquiries, Internal Logical Files, and External Interface Files
C. Lines, functions, modules, packages, projects
D. CPU, RAM, disk, network, display

Answer: B

Q903. What is the Halstead effort metric?

A. The physical effort of typing code
B. A measure derived from the number of operators and operands in code that estimates the mental effort required to develop or understand the program
C. The effort to install software
D. The effort to run tests

Answer: B

Q904. What is the purpose of a GQM (Goal-Question-Metric) approach in metrics?

A. A gaming qualification metric
B. To ensure metrics are aligned with specific goals by defining goals first, then questions to assess goal achievement, then metrics to answer questions
C. A graphics quality metric
D. A random metrics selection approach

Answer: B

Q905. What is the difference between subjective and objective metrics?

A. They are the same
B. Objective metrics are based on quantifiable measurements; subjective metrics involve human judgment and perception
C. Subjective metrics are always better
D. Objective metrics are always inaccurate

Answer: B

Q906. What is a Change Control Board (CCB)?

A. A circuit board for hardware changes
B. A group of stakeholders responsible for evaluating, approving, or rejecting proposed changes to configuration items
C. A board game about changes
D. A software testing board

Answer: B

Q907. What is the purpose of a build pipeline in configuration management?

A. Building a physical pipeline
B. An automated workflow that compiles, tests, and packages software whenever changes are committed to the repository
C. A water pipeline
D. An email pipeline

Answer: B

Q908. What is the difference between Git merge and Git rebase?

A. They are identical operations
B. Merge creates a merge commit preserving branch history; rebase moves commits to the tip of the target branch creating a linear history
C. Merge always causes conflicts
D. Rebase never modifies commit history

Answer: B

Q909. What is semantic versioning (SemVer)?

A. A versioning system based on how versions sound
B. A versioning scheme using MAJOR.MINOR.PATCH numbers where each indicates the type and scope of changes made
C. A random version numbering scheme
D. Versioning based on release dates

Answer: B

Q910. What is the purpose of a branching strategy?

A. A gardening technique
B. A set of rules defining how branches are created, named, and merged to organize parallel development and releases
C. A marketing strategy
D. A hiring strategy

Answer: B

Q911. What is GitFlow as a branching strategy?

A. A water flow monitoring tool
B. A branching model using main, develop, feature, release, and hotfix branches with defined rules for merging between them
C. A Git tutorial
D. A deployment tool

Answer: B

Q912. What is the purpose of configuration status accounting?

A. Tracking the company's finances
B. Recording and reporting the status of configuration items and change requests throughout the project lifecycle
C. Counting lines of code
D. Accounting for team hours

Answer: B

Q913. What is an artifact repository in CI/CD?

A. A museum for code artifacts
B. A centralized storage system for build outputs (compiled code, packages, containers) that can be versioned and deployed
C. A database for test results
D. A code hosting platform

Answer: B

Q914. What is a risk probability-impact matrix?

A. A mathematical equation
B. A grid that plots risks by their probability and impact to prioritize them and determine appropriate response strategies
C. A spreadsheet template
D. A type of decision tree

Answer: B

Q915. What are the common categories of software project risks?

A. Only technical risks exist
B. Technology, people, organizational, tools, requirements, and estimation risks among others
C. Only budget risks
D. Only schedule risks

Answer: B

Q916. What is risk exposure calculated as?

A. Risk probability minus risk impact
B. Risk probability multiplied by risk impact, representing the expected loss from the risk
C. Risk probability divided by risk impact
D. Risk probability plus risk impact

Answer: B

Q917. What is a risk trigger (or warning sign)?

A. A gun trigger
B. An event or condition that indicates a risk is about to occur or has occurred, prompting activation of the risk response plan
C. A software trigger
D. A database trigger

Answer: B

Q918. What is the concept of 'secondary risk'?

A. A less important risk
B. A new risk that arises as a direct result of implementing a risk response for a primary risk
C. The second risk identified
D. A backup risk plan

Answer: B

Q919. What is the purpose of a risk audit?

A. Auditing the company's finances
B. Examining and documenting the effectiveness of risk responses and the accuracy of the risk management process
C. Auditing employee performance
D. Auditing code quality

Answer: B

Q920. What is SQL injection and how is it prevented?

A. Injecting SQL databases with vitamins
B. An attack where malicious SQL code is inserted into application queries; prevented by using parameterized queries and input validation
C. A database optimization technique
D. A method for speeding up queries

Answer: B

Q921. What is the principle of defense in depth?

A. Building deep bunkers
B. Implementing multiple layers of security controls so that if one layer fails, others still provide protection
C. Using only one very strong password
D. Having one firewall

Answer: B

Q922. What is a session hijacking attack?

A. Stealing a classroom session plan
B. An attack where an attacker takes over a legitimate user's active session by stealing or predicting the session identifier
C. Hijacking a training session
D. Canceling a meeting

Answer: B

Q923. What is role-based access control (RBAC)?

A. Controlling roles in a theater play
B. A security model that restricts system access based on the roles assigned to users rather than individual user identities
C. A role-playing game
D. A hiring management system

Answer: B

Q924. What is the purpose of penetration testing?

A. Testing physical building penetration
B. Simulating real-world attacks against a system to identify vulnerabilities before malicious attackers can exploit them
C. Testing code penetration rates
D. Testing market penetration

Answer: B

Q925. What is a security token?

A. A subway token
B. A device or software that generates one-time passwords or cryptographic keys used for authenticating user identity
C. A casino chip
D. A gift card

Answer: B

Q926. What is the difference between a product backlog and a sprint backlog in Scrum?

A. They are the same thing used at different stages of the project
B. The product backlog contains all desired features for the product while the sprint backlog contains only items selected for the current sprint
C. The sprint backlog is maintained by management while the product backlog is maintained by developers
D. The product backlog is created once and never changes while the sprint backlog is updated daily

Answer: B

Q927. What is the Scrum of Scrums technique used for?

A. Replacing the daily standup meeting in small teams
B. Coordinating work across multiple Scrum teams working on the same product
C. Training new Scrum Masters on agile practices
D. Evaluating individual developer performance within a sprint

Answer: B

Q928. What is the Volere requirements template and what is its primary benefit?

A. A coding template that generates requirements automatically from source code
B. A structured template that provides a comprehensive framework for capturing all types of requirements systematically
C. A UML diagram type used only for non-functional requirements
D. A testing framework that validates requirements against code

Answer: B

Q929. How do quality function deployment (QFD) techniques apply to requirements engineering?

A. QFD is only used in manufacturing and has no application in software
B. QFD translates customer needs into technical requirements by mapping relationships between what customers want and how the system will deliver it
C. QFD replaces all forms of requirements elicitation with automated tools
D. QFD is a testing technique that validates requirements after implementation

Answer: B

Q930. What is the purpose of a RACI matrix in project management?

A. To calculate the project budget using mathematical formulas
B. To define roles and responsibilities by identifying who is Responsible, Accountable, Consulted, and Informed for each task
C. To schedule tasks on a calendar view similar to a Gantt chart
D. To track defects found during software testing

Answer: B

Q931. What is parametric estimation in software project management?

A. Estimation based entirely on the project manager's intuition and experience
B. Using statistical relationships between historical data and project parameters to calculate estimates
C. Asking each developer to guess how long their tasks will take
D. Copying the schedule from a previous project without any adjustments

Answer: B

Q932. What is the difference between a project baseline and a project plan?

A. They are the same document used at different project phases
B. A baseline is the approved version of the plan used to measure project performance against, while the plan is the working document that may be updated
C. A baseline is created after project completion while a plan is created before
D. A plan is used only for small projects while a baseline is used for large projects

Answer: B

Q933. What is the Builder design pattern used for?

A. Ensuring a class has only one instance throughout the application
B. Separating the construction of a complex object from its representation so the same construction process can create different representations
C. Defining a one-to-many dependency between objects so that changes in one notify all dependents
D. Providing a simplified interface to a complex subsystem

Answer: B

Q934. What is the Facade design pattern?

A. A pattern that adds new responsibilities to an object dynamically
B. A pattern that provides a simplified, unified interface to a set of interfaces in a subsystem
C. A pattern that allows incompatible interfaces to work together
D. A pattern that separates abstraction from implementation

Answer: B

Q935. What is the Interface Segregation Principle (ISP)?

A. All classes must implement every method defined in any interface they use
B. Clients should not be forced to depend on interfaces they do not use
C. Interfaces should contain as many methods as possible for reusability
D. Only one interface should exist per application module

Answer: B

Q936. What is the Iterator design pattern?

A. A pattern that converts the interface of a class into another interface clients expect
B. A pattern that provides a way to access elements of a collection sequentially without exposing its underlying representation
C. A pattern that defines a family of algorithms and makes them interchangeable
D. A pattern that composes objects into tree structures to represent part-whole hierarchies

Answer: B

Q937. What is the Broker architectural pattern?

A. A pattern where all requests go through a single centralized database
B. A pattern that decouples components by using a broker to coordinate communication between distributed services
C. A pattern that requires all components to be written in the same programming language
D. A pattern used exclusively for real-time embedded systems

Answer: B

Q938. What is the blackboard architectural pattern?

A. A pattern where developers write requirements on a physical whiteboard
B. A pattern where multiple specialized knowledge sources collaborate by reading from and writing to a shared data structure to solve complex problems
C. A pattern identical to the client-server model but with a different name
D. A pattern that restricts access to a single database to one user at a time

Answer: B

Q939. What is the F-pattern layout in UI design?

A. A layout that arranges all elements in a fixed grid pattern
B. A layout based on eye-tracking research showing users scan content in an F-shaped pattern, reading horizontally across the top then scanning vertically down the left side
C. A layout that uses only the letter F as a design element
D. A layout specifically designed for mobile-first applications

Answer: B

Q940. What is the purpose of user journey mapping?

A. To map the physical location of users using GPS technology
B. To visualize the complete experience a user has with a product from initial contact through engagement and into long-term use
C. To create a route map showing the shortest navigation path in the application
D. To document the travel expenses of the design team

Answer: B

Q941. What is the difference between a framework and a library in software implementation?

A. They are the same thing; the terms are used interchangeably
B. A framework calls your code (inversion of control) while a library is called by your code
C. A library is always larger than a framework in terms of file size
D. Frameworks are only used for web development while libraries are used for desktop applications

Answer: B

Q942. What is decision table testing?

A. A management technique for deciding which tests to skip
B. A black-box testing technique that uses a table to represent combinations of inputs and their corresponding expected outputs or actions
C. A method for testing database query performance
D. A way to schedule testing activities in a project timeline

Answer: B

Q943. What is the difference between a test driver and a test stub?

A. They are the same thing used at different project phases
B. A test driver calls the module under test (simulating the caller), while a test stub replaces a module called by the module under test (simulating the dependency)
C. A test driver is used for black-box testing while a test stub is used for white-box testing
D. A test driver runs tests automatically while a test stub requires manual execution

Answer: B

Q944. What is Lehman's Law of Continuing Change and its implications for software maintenance?

A. Software that is never used does not need maintenance
B. A system used in a real-world environment must be continually adapted or it becomes progressively less satisfactory to users
C. All software eventually becomes unmaintainable regardless of effort
D. Software maintenance costs decrease over time as the system stabilizes

Answer: B

Q945. What is the difference between prevention costs and appraisal costs in the cost of quality model?

A. Prevention costs and appraisal costs are the same category of quality costs
B. Prevention costs are spent on activities to prevent defects from occurring, while appraisal costs are spent on activities to detect defects that have already been introduced
C. Appraisal costs prevent defects while prevention costs detect them
D. Prevention costs only apply to hardware while appraisal costs only apply to software

Answer: B

Q946. How does the ISO/IEC 25010 quality model categorize software quality characteristics?

A. It defines only two quality characteristics: functionality and performance
B. It defines eight quality characteristics including functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability, and portability
C. It defines quality solely based on the number of defects per thousand lines of code
D. It categorizes quality only by user satisfaction survey scores

Answer: B

Q947. What is the depth of inheritance tree (DIT) metric and what does a high value indicate?

A. It measures the number of files in a project; a high value indicates a large project
B. It measures the maximum length from a class to the root of its inheritance hierarchy; a high value indicates greater complexity and potential for inherited behavior issues
C. It measures the depth of nested loops in code; a high value indicates faster execution
D. It measures the number of test cases per class; a high value indicates thorough testing

Answer: B

Q948. What is the fan-in and fan-out metric in software design?

A. Fan-in counts the number of developers who modified a module while fan-out counts the number of testers who tested it

B. Fan-in measures the number of modules that call a given module while fan-out measures the number of modules called by a given module

C. Fan-in measures input file size while fan-out measures output file size

D. Fan-in and fan-out are terms used only in electrical engineering, not software

Answer: B

Q949. What is the purpose of a deployment pipeline in configuration management?

A. To physically transport software on USB drives to client locations

B. To automate the process of building, testing, and deploying software through a series of stages from development to production

C. To create documentation for each software release

D. To assign developers to specific configuration management tasks

Answer: B

Q950. What is Infrastructure as Code (IaC) in the context of configuration management?

A. Writing application code that runs on infrastructure servers

B. Managing and provisioning computing infrastructure through machine-readable configuration files rather than manual processes

C. Using hardware components as programming languages

D. Coding directly on production servers without any version control

Answer: B

Q951. What is the RMMM (Risk Mitigation, Monitoring, and Management) plan?

A. A financial auditing plan for tracking project expenses

B. A comprehensive plan that documents risk mitigation strategies, monitoring activities, and management approaches for all identified project risks

C. A plan for managing team member vacation schedules

D. A security plan for preventing unauthorized access to the codebase

Answer: B

Q952. What is the difference between a risk and an issue in project management?

A. There is no difference; the terms are interchangeable

B. A risk is a potential future event that may or may not occur, while an issue is a current problem that has already materialized and needs immediate action

C. An issue is less important than a risk and can be ignored

D. Risks are always technical while issues are always organizational

Answer: B

Q953. What is the purpose of a risk-driven spiral model in software development?

A. To eliminate all risks before any coding begins

B. To use risk analysis as the primary driver for iteration planning, addressing the highest risks earliest through prototyping and analysis

C. To create spiral-shaped project documentation

D. To assign each developer a specific risk to manage independently

Answer: B

Q954. What are technology risks in software projects?

A. Risks that only affect non-technical project activities

B. Risks related to the use of unproven technologies, technical complexity, platform dependencies, or performance uncertainties

C. Risks that are always impossible to mitigate

D. Risks that only occur during the maintenance phase of a project

Answer: B

Q955. What is the purpose of threat modeling in software security engineering?

A. To create 3D models of physical security threats to the building
B. To systematically identify, enumerate, and prioritize potential security threats to a system so appropriate countermeasures can be designed
C. To model the performance characteristics of the software under load
D. To create user interface mockups that look secure to end users

Answer: B

Q956. What is cross-site request forgery (CSRF) and how is it prevented?

A. An attack where multiple websites share the same database and it cannot be prevented
B. An attack that tricks a user's browser into making unwanted requests to a trusted site, prevented using anti-CSRF tokens and same-site cookie attributes
C. An attack that only targets mobile applications and is prevented by app store reviews
D. An encryption algorithm used to protect data in transit

Answer: B

Q957. What is the CIA triad in information security?

A. A government intelligence agency classification system for software projects
B. Three core principles of information security: Confidentiality (data accessible only to authorized parties), Integrity (data is accurate and unaltered), and Availability (data is accessible when needed)
C. A project management framework with three phases: Create, Implement, Archive
D. A testing methodology using three levels of test cases

Answer: B

Q958. What is the difference between authentication and authorization in security?

A. They are the same process and can be used interchangeably
B. Authentication verifies the identity of a user (who they are), while authorization determines what actions or resources the authenticated user is permitted to access (what they can do)
C. Authorization happens before authentication in all security systems
D. Authentication is used only for web applications while authorization is used only for mobile applications

Answer: B

Q959. What is the purpose of an assertion in software implementation?

A. To display output to the user interface
B. To verify that a condition the programmer believes to be true actually holds at a specific point in the code, catching logical errors early
C. To encrypt sensitive data before storing it in the database
D. To create a backup of the source file before compilation

Answer: B

Q960. What is the Weighted Methods per Class (WMC) metric in object-oriented design?

A. A metric that counts the total weight of class files in kilobytes
B. The sum of complexities of all methods defined in a class, indicating the class's overall complexity and effort required to develop and maintain it
C. The average number of parameters across all methods in a class
D. A metric that measures how many classes inherit from a given class

Answer: B

Hard Questions

480 questions

Q961. Which of the following best describes 'socio-technical systems'?

A. Systems that include both technical and organizational components
B. Systems that contain only technical components
C. Web-based social media communication platforms
D. Internal technical documentation management systems

Answer: A

Q962. What is Lehman's first law of software evolution?

A. Software systems must continually grow in size or become obsolete in the system context
B. The inherent complexity of software always decreases over its lifetime during implementation
C. Well-designed software systems never require updates or modifications at all for quality purposes
D. A program used in a real-world environment must continually change or become progressively less useful

Answer: D

Q963. In the context of software engineering ethics, what does the ACM/IEEE Code of Ethics emphasize?

A. Public interest and professional responsibility
B. Rapid speed of software delivery timelines
C. Using the minimum documentation possible
D. Maximizing project profits above all else

Answer: A

Q964. What is the key challenge addressed by Brooks' 'No Silver Bullet' paper?

A. Network bandwidth and throughput constraints by the organization
B. A critical lack of modern programming languages as a standard approach
C. Fundamental hardware performance limitations in practice typically
D. Essential complexity of software cannot be eliminated by any single technique

Answer: D

Q965. Which concept distinguishes software engineering from ad hoc programming?

A. Systematic, disciplined, and quantifiable approach
B. Simply writing more lines of source code
C. Using a standard keyboard for input tasks
D. Utilizing more recent hardware component sets

Answer: A

Q966. What is the 'wicked problem' concept in software engineering?

A. A straightforward and simple coding bug fix across all phases
B. A problem that is difficult to define and has no definitive solution
C. A software problem caused by a virus attack at every stage
D. A recurring hardware malfunction issue type in a systematic way

Answer: B

Q967. What is the primary difference between software engineering and systems engineering?

A. Software engineering also includes hardware design within the system boundary
B. Systems engineering exclusively covers testing by the development process
C. The two disciplines are entirely identical for the project goals according to best practices
D. Systems engineering covers the entire system including hardware; SE focuses on software

Answer: D

Q968. What does 'accidental complexity' mean in software engineering?

A. Unexpected failures in hardware components as defined by standards
B. Complexity introduced by the tools and methods used, not inherent to the problem
C. Random and unpredictable bugs in source code over the entire lifecycle
D. Inherent complexity from the problem domain and its related activities

Answer: B

Q969. Which of the following represents a key principle of software engineering according to Sommerville?

A. Always maximize the total length of source code
B. Systems should be developed using a managed and understood process
C. Avoid writing any form of technical documentation
D. Use only one single programming language always

Answer: B

Q970. What role does abstraction play in software engineering?

A. It increases the total length of the code
B. It eliminates the need for software testing
C. It helps manage complexity by hiding unnecessary details
D. It completely removes all project documentation

Answer: C

Q971. What is the Rational Unified Process (RUP)?

A. An iterative framework with four phases: Inception, Elaboration, Construction, Transition
B. A basic and simple coding technique only as defined by standards for all project stakeholders
C. A relational database design model type in all development efforts by the project team members
D. A specialized software testing tool suite within the given constraints

Answer: A

Q972. What distinguishes the concurrent development model from other models?

A. It only supports a single developer at a time to achieve project objectives
B. All activities occur simultaneously with states rather than sequential phases
C. It completely eliminates all development phases and related components
D. It has absolutely no testing activities at all in the engineering discipline

Answer: B

Q973. In the Spiral model, what happens if a risk cannot be mitigated?

A. Skip directly to the deployment phase
B. Continue development regardless of the risk
C. Ignore the risk and move forward
D. The project may be terminated

Answer: D

Q974. What is the Personal Software Process (PSP)?

A. A hardware development process model type as part of the methodology
B. A standard network communication protocol in the development process
C. A team management and organizational method for effective project outcomes
D. A structured framework for individual developers to measure and improve their work

Answer: D

Q975. What is the Team Software Process (TSP)?

A. A method designed for a single developer only during the software lifecycle
B. A specific type of software testing process within the project scope
C. A hardware design and manufacturing method for the development team
D. A framework that builds on PSP to guide teams in developing software-intensive products

Answer: D

Q976. What does CMMI stand for and what is its purpose?

A. Capability Maturity Model Integration — a process improvement framework
B. Central Management Module Index for projects in the system context
C. Computer Made Machine Interface for hardware throughout the project
D. Code Maintenance Manual Integration standard during implementation

Answer: A

Q977. What are the five maturity levels of CMMI?

A. Initial, Managed, Defined, Quantitatively Managed, Optimizing
B. Code, Test, Deploy, Monitor, Maintain in practice typically
C. Beginner, Intermediate, Advanced, Expert, Master
D. Plan, Do, Check, Act, Review for quality purposes

Answer: A

Q978. What is the key difference between prescriptive and adaptive process models?

A. Prescriptive models do not involve any planning at every stage across all phases
B. Adaptive models typically require more documentation by the organization
C. There is absolutely no real difference between them as a standard approach
D. Prescriptive models define a fixed set of activities; adaptive models adjust based on feedback

Answer: D

Q979. What is the Cleanroom software engineering approach?

A. A formal methods approach emphasizing defect prevention over defect removal
B. A hardware testing and validation environment according to best practices
C. A physical room dedicated to coding activities for the project goals
D. A model focused on clean code formatting style in a systematic way

Answer: A

Q980. In the context of process models, what is a 'process pattern'?

A. A software design pattern for code modules within the system boundary
B. An individual test case for code validation and its related activities
C. A specific coding style guideline for teams by the development process
D. A template describing a proven process solution to a recurring problem

Answer: D

Q981. What is the Scaled Agile Framework (SAFe)?

A. A framework for scaling Agile practices to enterprise level
B. A framework designed for individual small teams
C. A comprehensive testing automation framework tool
D. An established coding standard for all projects

Answer: A

Q982. What is the difference between Scrum and Kanban?

A. Scrum uses fixed-length sprints; Kanban uses continuous flow with WIP limits
B. Kanban uses fixed-length sprints; Scrum does not across all phases
C. The two methodologies are completely identical at every stage
D. Scrum has no defined roles; Kanban has defined roles in a systematic way

Answer: A

Q983. What is 'technical debt' in Agile context?

A. Physical hardware equipment purchase expenses according to best practices
B. The cost of rework caused by choosing quick solutions over better approaches
C. Annual recurring software licensing fee costs within the system boundary
D. Financial money owed to the development team for the project goals

Answer: B

Q984. What is the Crystal methodology family?

A. A family of Agile methodologies sized by team size and criticality
B. A specialized software testing automation tool by the development process
C. A general-purpose programming language format and its related activities
D. A relational database management system design over the entire lifecycle

Answer: A

Q985. What is Feature-Driven Development (FDD)?

A. A random unstructured development approach only as defined by standards
B. A comprehensive documentation standard and format within the given constraints
C. An Agile method that organizes development around building features in two-week iterations
D. A formal software testing methodology and approach for all project stakeholders

Answer: C

Q986. In Agile estimation, what is 'Planning Poker'?

A. A recreational team card game activity only in all development efforts
B. A project planning and scheduling tool suite by the project team members
C. A formal risk analysis assessment method type and related components
D. A consensus-based estimation technique using cards with numbers

Answer: D

Q987. What is the purpose of a 'Burndown Chart'?

A. To track overall project financial expenses
B. To show remaining work versus time in a sprint or project
C. To measure and assess overall code quality
D. To track daily employee attendance records

Answer: B

Q988. What is Dynamic Systems Development Method (DSDM)?

A. An Agile framework that fixes time and cost, varying features to meet business needs
B. A standard network communication protocol type for effective project outcomes
C. A hardware design and manufacturing method only to achieve project objectives
D. A relational database design methodology approach in the engineering discipline

Answer: A

Q989. What does 'MoSCoW' prioritization stand for?

A. Monitor, Observe, Scan, Check, Optimize, Watch
B. Must have, Should have, Could have, Won't have
C. A Russian city
D. Manage, Organize, Schedule, Control, Outline, Work

Answer: B

Q990. What is the concept of 'Emergent Design' in Agile?

A. Designing the entire system completely upfront first
B. Allowing the design to evolve incrementally as understanding grows
C. Having absolutely no design process at all ever
D. Copying designs from existing similar systems only

Answer: B

Q991. What is the Volere requirements template?

A. A software testing automation template setup within the project scope
B. A comprehensive template for organizing and documenting requirements with quality gateways
C. A deployment configuration setup template for the development team throughout the project
D. A source code implementation template file during the software lifecycle

Answer: B

Q992. What is goal-oriented requirements engineering?

A. An approach where requirements are derived from stakeholder and system goals
B. Setting performance and speed benchmarks for quality purposes
C. Setting rigid project deadline target dates in the system context
D. Writing only goals instead of actual code during implementation

Answer: A

Q993. What is the KAOS methodology?

A. A deployment and release methodology guide by the organization
B. A goal-oriented requirements engineering methodology using formal methods
C. A specific software testing methodology only as a standard approach
D. A random unstructured development approach in practice typically

Answer: B

Q994. What is requirements volatility?

A. Requirements that remain perfectly stable always at every stage
B. A type of volatile computer memory hardware across all phases
C. A recurring issue with hardware component sets in a systematic way
D. The tendency of requirements to change during the development process

Answer: D

Q995. What is viewpoint-oriented requirements engineering?

A. Collecting requirements from multiple stakeholder viewpoints to ensure completeness
B. Only considering developer technical viewpoints within the system boundary
C. Deliberately ignoring all stakeholder views given according to best practices
D. Using only a single stakeholder perspective view for the project goals

Answer: A

Q996. What is the difference between 'shall' and 'should' in requirements documents?

A. 'Shall' indicates mandatory requirements; 'should' indicates desirable but optional
B. The two terms mean exactly the same thing always by the development process
C. Neither term is used in requirements documents ever over the entire lifecycle
D. 'Should' is mandatory and 'shall' is always optional and its related activities

Answer: A

Q997. What is formal specification in requirements engineering?

A. Drawing informal pictures of system requirements
B. Using mathematical notation to precisely define system requirements
C. Using programming code as the requirements source
D. Writing requirements in an informal manner only

Answer: B

Q998. What are cross-cutting concerns in requirements?

A. Requirements that are very easy to implement fast as defined by standards
B. Requirements that are considered not important for all project stakeholders
C. Requirements that affect multiple modules and cannot be cleanly decomposed
D. Requirements from only one single stakeholder within the given constraints

Answer: C

Q999. What is the i* (i-star) framework?

A. A numerical system for rating software products in all development efforts
B. A goal-oriented and agent-oriented modeling framework for requirements engineering
C. A general-purpose application programming framework by the project team members
D. A comprehensive software testing automation framework and related components

Answer: B

Q1000. What is the difference between derived and imposed requirements?

A. Derived requirements are always considered optional in the development process during the software lifecycle
B. Derived requirements come from higher-level requirements; imposed requirements come from external constraints
C. They are fundamentally the same concept entirely to achieve project objectives in the engineering discipline
D. Imposed requirements are always derived from code for effective project outcomes as part of the methodology

Answer: B

Q1001. What is the COCOMO II model's key improvement over COCOMO I?

A. It uses significantly simpler mathematical formulas within the system boundary
B. It provides much faster calculation processing speed for the project goals
C. It produces much better visual graphics output according to best practices
D. It accounts for modern development practices like reuse and object-oriented development

Answer: D

Q1002. What is the Delphi estimation technique?

A. A specific programming implementation technique only
B. A comprehensive software testing methodology type
C. A production deployment automation technique type
D. An expert-based estimation method using anonymous iterative rounds

Answer: D

Q1003. What is the Putnam model (SLIM)?

A. A personal health and diet improvement plan only by the development process
B. A comprehensive software testing process model type and its related activities
C. A software architecture and design model for teams over the entire lifecycle
D. A software estimation model relating effort to delivery time using the Rayleigh curve

Answer: D

Q1004. What is the difference between bottom-up and top-down estimation?

A. Top-down estimation is always much more accurate in all development efforts by the project team members
B. Bottom-up estimation is always significantly faster within the given constraints
C. Bottom-up estimates individual tasks then aggregates; top-down estimates the whole project then decomposes
D. Both approaches always give identical final results as defined by standards for all project stakeholders

Answer: C

Q1005. What is the concept of 'Brooks' Law'?

A. Adding people to a late software project makes it later
B. A specific programming coding law and its rules
C. Established laws about code formatting standards
D. More developers always mean faster project work

Answer: A

Q1006. What is the Schedule Performance Index (SPI) in EVM?

A. A comprehensive software testing progress index in the engineering discipline
B. A specific source code quality coding metric only to achieve project objectives
C. The ratio of earned value to planned value, indicating schedule efficiency
D. A generic system processing speed measurement and related components

Answer: C

Q1007. What is the Cost Performance Index (CPI) in EVM?

A. A general consumer price comparison index only for effective project outcomes
B. A specific source code quality coding metric only as part of the methodology
C. The ratio of earned value to actual cost, indicating cost efficiency
D. A hardware system performance rating metric in the development process

Answer: C

Q1008. What is Monte Carlo simulation used for in project management?

A. A recreational casino gambling game activity only during the software lifecycle
B. Simulating project outcomes using probability distributions to assess risk
C. A specific software coding implementation technique within the project scope
D. A software architecture and design method guide for the development team

Answer: B

Q1009. What is the concept of 'crashing' in project scheduling?

A. An unplanned system failure or complete breakdown
B. Adding resources to critical path tasks to reduce project duration
C. Permanently removing tasks from the project plan
D. Completely cancelling the entire project scope

Answer: B

Q1010. What is fast tracking in project management?

A. Reducing the total number of project requirements during implementation
B. Requiring the team to work overtime every day throughout the project
C. Skipping all software testing activities entirely in the system context
D. Performing sequential activities in parallel to compress the schedule

Answer: D

Q1011. What is the SOLID acronym in object-oriented design?

A. Single Responsibility, Open/Closed, Liskov Substitution, Interface Segregation, Dependency Inversion
B. A comprehensive software testing framework set for effective project outcomes
C. A specific general-purpose coding language type in the engineering discipline
D. A classification of physical building materials and related components to achieve project objectives

Answer: A

Q1012. What is the Dependency Inversion Principle?

A. All dependencies should always point downward only as part of the methodology
B. All module dependencies should be completely removed in the development process
C. High-level modules should not depend on low-level modules; both should depend on abstractions
D. Dependencies between modules should be circular during the software lifecycle

Answer: C

Q1013. What is the Interface Segregation Principle?

A. Avoid defining all interfaces in the system entirely
B. Use one single large monolithic interface always
C. Clients should not be forced to depend on interfaces they don't use
D. Combine all interfaces into a single large one

Answer: C

Q1014. What is the Decorator design pattern?

A. A pattern that attaches additional responsibilities to an object dynamically
B. A user interface visual styling pattern only within the project scope
C. A relational database schema design pattern type for the development team
D. A comprehensive software testing design pattern throughout the project

Answer: A

Q1015. What is the difference between composition and inheritance?

A. The two concepts are completely identical always in the system context
B. Inheritance is never used in modern source code for quality purposes
C. Composition is always the wrong approach to use during implementation
D. Inheritance creates an 'is-a' relationship; composition creates a 'has-a' relationship

Answer: D

Q1016. What is the Adapter design pattern?

A. A comprehensive software testing adapter tool
B. A relational database connection driver and tool
C. A physical electrical power adapter device unit
D. A pattern that allows incompatible interfaces to work together

Answer: D

Q1017. What is the Command design pattern?

A. A pattern that encapsulates a request as an object, allowing parameterization and queuing
B. A production deployment automation command set by the organization at every stage
C. A command-line interface terminal command only in practice typically
D. A comprehensive software testing command suite as a standard approach

Answer: A

Q1018. What is the Template Method pattern?

A. A source code boilerplate template file format across all phases in a systematic way
B. A software testing test case plan template set according to best practices
C. A comprehensive project documentation template for the project goals
D. A pattern that defines the skeleton of an algorithm, deferring some steps to subclasses

Answer: D

Q1019. What is the concept of 'design debt'?

A. Financial debt for purchasing design tools only within the system boundary
B. The accumulated cost of design shortcuts that will require future rework
C. A specific type of design pattern category name by the development process
D. A concept related only to software testing work and its related activities

Answer: B

Q1020. What is the difference between architectural design and detailed design?

A. They are fundamentally identical concepts entirely over the entire lifecycle as defined by standards
B. Architectural design only involves UI layout work in all development efforts by the project team members
C. Detailed design always comes before architectural for all project stakeholders within the given constraints
D. Architectural design defines system structure; detailed design specifies algorithms and data structures within components

Answer: D

Q1021. What is the TOGAF framework?

A. The Open Group Architecture Framework for enterprise architecture
B. A production deployment automation framework for effective project outcomes
C. A general-purpose source code framework tool to achieve project objectives
D. A comprehensive software testing framework type in the engineering discipline

Answer: A

Q1022. What is the Architecture Tradeoff Analysis Method (ATAM)?

A. A method for evaluating architectural decisions against quality attribute requirements
B. A mathematical formula for doing calculations as part of the methodology
C. A comprehensive software testing methodology during the software lifecycle
D. A specific source code writing methodology only in the development process

Answer: A

Q1023. What is the concept of 'architectural erosion'?

A. The gradual degradation of architecture as implementation deviates from the intended design
B. A specific category of reported software defect for the development team
C. A concept related to software testing only throughout the project in the system context
D. Physical decay of building materials over time within the project scope

Answer: A

Q1024. What is a Domain-Specific Software Architecture (DSSA)?

A. A generic all-purpose architecture template type during implementation
B. A network infrastructure architecture blueprint in practice typically
C. An architecture tailored to a specific application domain with reusable components
D. A comprehensive software testing architecture for quality purposes

Answer: C

Q1025. What is the CQRS pattern?

A. A specific source code implementation pattern by the organization
B. A standard database query language command type as a standard approach
C. A comprehensive software testing pattern type at every stage
D. Command Query Responsibility Segregation — separating read and write operations

Answer: D

Q1026. What is the Strangler Fig pattern?

A. A migration pattern that incrementally replaces a legacy system by gradually routing traffic to a new system
B. An application security protection pattern type within the system boundary by the development process
C. A comprehensive software testing pattern approach for the project goals according to best practices
D. A botanical plant species classification category across all phases in a systematic way

Answer: A

Q1027. What is the concept of 'fitness functions' in evolutionary architecture?

A. Physical exercise routines for the team members and its related activities
B. User interface design metrics and score values as defined by standards
C. System performance benchmarks and baselines set over the entire lifecycle
D. Automated tests that verify architectural characteristics are maintained

Answer: D

Q1028. What is the hexagonal architecture (ports and adapters)?

A. A building shaped with six equal sides design for all project stakeholders
B. A relational database schema design method type within the given constraints
C. An architecture that isolates the core logic from external concerns using ports and adapters
D. A network topology and routing structure layout in all development efforts

Answer: C

Q1029. What is the difference between architectural patterns and architectural styles?

A. Styles are more specific than patterns overall in the engineering discipline
B. Patterns are broader in scope than styles are and related components to achieve project objectives
C. Styles are broader design philosophies; patterns are specific solutions to recurring problems
D. The two concepts are completely identical overall by the project team members

Answer: C

Q1030. What is the concept of 'Conway's Law' in software architecture?

A. A specific legal regulatory requirement and law for effective project outcomes
B. A standard source code formatting law and rule as part of the methodology
C. A comprehensive software testing principle type in the development process
D. Organizations design systems that mirror their own communication structure

Answer: D

Q1031. What is the difference between formative and summative usability evaluation?

A. Formative evaluation only occurs after launch date within the system boundary
B. The two types of evaluation are the same thing according to best practices
C. Summative evaluation occurs during development by the development process
D. Formative evaluates during development to improve; summative evaluates the finished product

Answer: D

Q1032. What is cognitive load theory in UI design?

A. A database theory about query optimization work within the given constraints in all development efforts
B. A programming concept about memory management as defined by standards for all project stakeholders
C. The theory that users have limited mental processing capacity, and designs should minimize unnecessary cognitive effort
D. Brain scanning technology for development teams and its related activities over the entire lifecycle

Answer: C

Q1033. What is the Gestalt principle of proximity?

A. Physical distance between computer workstations by the project team members
B. A security principle about access proximity level to achieve project objectives
C. A networking concept about node proximity range and related components
D. Elements placed close together are perceived as belonging to the same group

Answer: D

Q1034. What is WCAG?

A. Web Content Accessibility Guidelines — standards for making web content accessible
B. A creational object-oriented design pattern type as part of the methodology
C. A comprehensive software testing framework tool for effective project outcomes
D. A general-purpose programming language type only in the engineering discipline

Answer: A

Q1035. What is the concept of 'dark patterns' in UI design?

A. Security patterns for data protection only within the project scope
B. Night mode designs for reducing eye strain during the software lifecycle
C. Dark color themes for user interface screens in the development process
D. Deceptive design practices that trick users into unintended actions

Answer: D

Q1036. What is card sorting in UX research?

A. A method where users organize topics into categories to inform information architecture
B. Sorting records stored in a database table only throughout the project
C. Sorting a deck of standard playing cards only for the development team
D. A specific software testing automation technique in the system context

Answer: A

Q1037. What is the System Usability Scale (SUS)?

A. A performance scale for rating hardware speed in practice typically
B. A standardized questionnaire for measuring perceived usability of a system
C. A musical scale for composing melody pieces during implementation
D. A weight scale for measuring device components for quality purposes

Answer: B

Q1038. What is the difference between learnability and memorability in usability?

A. Learnability is considered less important overall for the project goals according to best practices
B. Memorability always comes first in any evaluation at every stage across all phases in a systematic way
C. The two concepts are fundamentally the same thing as a standard approach by the organization
D. Learnability is how easy it is to learn initially; memorability is how easy it is to reuse after a period of not using

Answer: D

Q1039. What is the concept of 'information scent' in UI design?

A. Visual and textual cues that help users predict what they'll find by following a link or path
B. Data encryption and protection techniques used by the development process
C. The physical fragrance of a commercial product within the system boundary
D. File compression and archiving algorithms used and its related activities

Answer: A

Q1040. What is the difference between user-centered design (UCD) and activity-centered design (ACD)?

A. User-centered design completely ignores all users for all project stakeholders
B. UCD focuses on user needs and preferences; ACD focuses on the activities users perform
C. Activity-centered design is entirely obsolete now as defined by standards
D. The two approaches are completely identical overall over the entire lifecycle

Answer: B

Q1041. What is the concept of 'clean code'?

A. Code that has absolutely no comments at all as defined by standards
B. Source code that has been fully minimized down for all project stakeholders
C. Source code that has been fully obfuscated over within the given constraints
D. Code that is easy to read, understand, and maintain, following best practices

Answer: D

Q1042. What is the Boy Scout Rule in coding?

A. A comprehensive software testing protocol type
B. Write source code in an outdoor setting always
C. Always leave the code cleaner than you found it
D. A recreational camping and outdoors activity rule

Answer: C

Q1043. What is the Law of Demeter?

A. A relational database normalization rule set type and related components
B. A fundamental law of classical physics principles by the project team members
C. An ancient Greek legal and civil law code rule in all development efforts
D. A design guideline stating that an object should only talk to its immediate friends

Answer: D

Q1044. What is technical debt and how does it relate to implementation?

A. Annual recurring software licensing fee cost totals for effective project outcomes
B. The cost of shortcuts in implementation that will need to be fixed later
C. Monetary financial debt owed by a company entity to achieve project objectives
D. Physical hardware equipment purchase cost amounts in the engineering discipline

Answer: B

Q1045. What is the difference between defensive programming and offensive programming?

A. Offensive programming is always incorrect to use during the software lifecycle
B. The two approaches are completely identical overall in the development process
C. Defensive handles errors gracefully; offensive fails fast to expose bugs early
D. Sports programming for athletic events coverage as part of the methodology

Answer: C

Q1046. What is code obfuscation?

A. A specific debugging and tracing technique tool for the development team
B. A comprehensive software testing method approach throughout the project
C. Deliberately making code difficult to understand to prevent reverse engineering
D. Making source code more readable for developers within the project scope

Answer: C

Q1047. What is the concept of 'separation of concerns' in implementation?

A. Separating test code from production code files during implementation
B. Organizing code so each section addresses a distinct concern or functionality
C. Physical separation of developer workstations area in the system context
D. Separating frontend code from backend code only for quality purposes

Answer: B

Q1048. What is a design smell versus a code smell?

A. Design smells are always much worse than code smells as a standard approach
B. The two concepts are completely identical and same in practice typically
C. Code smells are always much worse than design smells by the organization
D. Code smells are implementation-level issues; design smells are higher-level structural problems

Answer: D

Q1049. What is the 'broken window theory' in software development?

A. A user interface visual design theory and concept in a systematic way
B. A theory about physical broken windows in homes at every stage across all phases
C. Neglecting small code quality issues leads to progressively worse code quality over time
D. A comprehensive software testing theory approach for the project goals

Answer: C

Q1050. What is the principle of least astonishment in programming?

A. Deliberately surprising users with behavior changes according to best practices
B. Minimizing features for simplicity above all else within the system boundary
C. A component should behave in a way that most users expect it to behave
D. Maximizing performance above all other attributes by the development process

Answer: C

Q1051. What is mutation testing?

A. Performing genetic and biological testing samples
B. Testing software for computer virus threats only
C. Introducing small changes to code to check if tests can detect them
D. Testing for mutations in stored database data only

Answer: C

Q1052. What is the difference between structural and functional testing?

A. Structural testing completely ignores the code base during implementation for quality purposes
B. Functional testing examines internal code structure throughout the project in the system context
C. Structural testing is based on internal code structure; functional testing is based on specifications
D. The two approaches are fundamentally the same thing for the development team

Answer: C

Q1053. What is path testing?

A. Testing all possible execution paths through the code
B. Testing GPS navigation system paths and routes
C. Testing operating system file paths and path names
D. Testing network routing paths and protocol flows

Answer: A

Q1054. What is cyclomatic complexity and how does it relate to testing?

A. A metric related to bicycle complexity and speed in practice typically as a standard approach
B. A relational database complexity metric and score by the organization at every stage
C. A code complexity metric that indicates the minimum number of test cases needed for full branch coverage
D. A network topology complexity metric and rating across all phases in a systematic way

Answer: C

Q1055. What is fuzz testing?

A. Testing using fuzzy logic-based algorithm systems for the project goals
B. Providing random, invalid, or unexpected data as input to find vulnerabilities
C. Testing audio and sound reproduction quality level within the system boundary
D. Testing with fuzzy and blurred test image inputs according to best practices

Answer: B

Q1056. What is the concept of 'test coverage criteria'?

A. The total file size of all test script source files
B. Rules that define what aspects of the software must be tested
C. The total number of testers on the project team
D. Simply how many test scripts currently exist total

Answer: B

Q1057. What is property-based testing?

A. Testing real estate management software systems by the development process
B. Testing that specified properties or invariants hold for all valid inputs
C. Testing CSS stylesheet property values and styles and its related activities
D. Testing object property and getter method values over the entire lifecycle

Answer: B

Q1058. What is the pesticide paradox in software testing?

A. Testing agricultural management software systems within the given constraints
B. Running the same tests repeatedly will eventually stop finding new defects
C. Using pesticides on actual computer bugs and pests as defined by standards
D. A chemical testing methodology approach and method for all project stakeholders

Answer: B

Q1059. What is the difference between fault, error, and failure?

A. The three concepts are fundamentally the same thing in all development efforts
B. An error is always immediately visible to all users to achieve project objectives
C. A fault is a defect in code; an error is an incorrect state; a failure is observable incorrect behavior
D. A failure is what always causes a fault to happen by the project team members and related components

Answer: C

Q1060. What is model-based testing?

A. Testing fashion model runway performance quality in the engineering discipline
B. Generating test cases automatically from a model of the system's behavior
C. Testing relational database data model structures for effective project outcomes
D. Testing three-dimensional rendering model objects as part of the methodology

Answer: B

Q1061. What is Lehman's Law of Self-Regulation?

A. Software evolution processes are self-regulating with statistically determinable trends
B. Software automatically regulates itself without help in practice typically
C. Programs never need any updates or patches applied as a standard approach
D. Developers regulate themselves without any guidance by the organization

Answer: A

Q1062. What is the concept of 'software archaeology'?

A. Digging up old physical computer equipment devices at every stage across all phases
B. A museum exhibit about computing history artifacts in a systematic way for the project goals
C. Physical hardware recovery and data retrieval work according to best practices
D. The study and recovery of knowledge about legacy systems through code analysis and documentation

Answer: D

Q1063. What is the difference between wrapping and re-engineering legacy systems?

A. Re-engineering is always significantly faster overall and its related activities
B. Wrapping is always significantly more expensive work by the development process
C. The two approaches are completely identical overall within the system boundary
D. Wrapping adds a new interface around the legacy system; re-engineering restructures its internals

Answer: D

Q1064. What is the concept of 'feature interaction' in maintenance?

A. Comprehensive feature testing and validation plans for all project stakeholders
B. Features physically meeting one another in person over the entire lifecycle
C. Formal feature documentation and specification docs as defined by standards
D. Unexpected behavior when independently developed features conflict or interfere

Answer: D

Q1065. What is the ripple effect in software maintenance?

A. Physical water ripples in a pond surface area within the given constraints
B. A user interface animation and transition effect by the project team members
C. Audio sound effects in the user interface design in all development efforts
D. The propagation of changes through the system, causing unintended side effects

Answer: D

Q1066. What is the Lehman-Belady model of software evolution?

A. A creational object-oriented design pattern type to achieve project objectives
B. A comprehensive software testing model and plan in the engineering discipline
C. A purely mathematical model for calculation work and related components
D. A set of laws describing the behavior and evolution of large software systems

Answer: D

Q1067. What is technical debt in the context of maintenance?

A. Accumulated shortcuts and suboptimal decisions that increase future maintenance cost
B. Financial debt for purchasing hardware equipment for effective project outcomes
C. Monthly developer salary payments and wage costs as part of the methodology
D. Annual recurring software licensing fee cost totals in the development process

Answer: A

Q1068. What is the concept of 'design recovery' in maintenance?

A. Using reverse engineering to reconstruct higher-level design abstractions from code
B. Recovering from catastrophic design errors quickly within the project scope
C. Recovering accidentally deleted design documents during the software lifecycle
D. Creating design backup copies and archive storage for the development team

Answer: A

Q1069. What is software rejuvenation?

A. Periodically restarting or reinitializing software to prevent degradation from accumulated errors
B. Making software visually look brand new and fresh throughout the project in the system context
C. Completely rewriting the entire software system over during implementation
D. Updating only the user interface visual design look for quality purposes in practice typically

Answer: A

Q1070. What is the relationship between maintainability and other quality attributes?

A. Maintainability is completely independent of everything at every stage
B. Maintainability often trades off with performance but supports reliability and portability
C. There is absolutely no relationship at all between them as a standard approach
D. Maintainability conflicts with all other quality attributes by the organization

Answer: B

Q1071. What is the GQM (Goal-Question-Metric) approach?

A. A recreational video gaming strategy method approach in all development efforts by the project team members
B. A source code implementation coding approach method and related components to achieve project objectives
C. A systematic approach to defining goals, deriving questions, and identifying metrics for quality measurement
D. A comprehensive software testing approach method only in the engineering discipline

Answer: C

Q1072. What is Statistical Process Control (SPC) in software QA?

A. A source code implementation coding technique method as part of the methodology
B. A software architecture and design method guide book in the development process
C. Using statistical methods to monitor and control software processes to ensure they operate predictably
D. A statistical report generated for project managers for effective project outcomes

Answer: C

Q1073. What is the concept of 'process maturity' in QA?

A. The overall size of the development process and scope for the development team throughout the project
B. Processes that have become outdated and old over time during the software lifecycle
C. The execution speed of all development processes running within the project scope
D. The extent to which an organization's processes are defined, managed, measured, and continuously improved

Answer: D

Q1074. What is Root Cause Analysis (RCA)?

A. A systematic process for identifying the underlying causes of defects or problems
B. Finding the roots of a physical plant in soil in the system context
C. A production deployment automation process method for quality purposes
D. A source code implementation coding technique only during implementation

Answer: A

Q1075. What is the difference between product quality and process quality?

A. Product quality is entirely not measurable or assessable by the organization
B. The two concepts are completely identical to each other in practice typically
C. Product quality measures the software itself; process quality measures the development process
D. Process quality is completely irrelevant to all outcomes as a standard approach

Answer: C

Q1076. What is a causal analysis meeting?

A. An informal casual meeting with team colleagues at every stage
B. A meeting to analyze the root causes of defects and identify preventive actions
C. A recreational team building social activity event in a systematic way
D. A project planning and scheduling meeting session across all phases

Answer: B

Q1077. What is the concept of 'quality gates' in software development?

A. A type of network firewall configuration and setup according to best practices
B. Physical security gates at building entrance areas for the project goals
C. Checkpoints in the development process where quality criteria must be met before proceeding
D. Security gates for data access control setup only within the system boundary

Answer: C

Q1078. What is the Clean Room approach to quality?

A. A physically tidy and organized workspace and area by the development process
B. A hardware manufacturing clean room facility space and its related activities
C. A software development approach using formal verification to prevent defects rather than test for them
D. A specific testing environment setup and space area over the entire lifecycle

Answer: C

Q1079. What is the relationship between CMMI and software quality?

A. CMMI only measures physical hardware quality metrics for all project stakeholders
B. There is absolutely no relationship between the two as defined by standards
C. CMMI provides a framework for improving processes, which directly impacts software quality
D. CMMI completely replaces all software testing types within the given constraints

Answer: C

Q1080. What is the concept of 'zero defect' philosophy?

A. Deliberately ignoring all reported defect reports and related components
B. Having absolutely no tests in the entire project by the project team members
C. A quality approach emphasizing prevention and doing things right the first time
D. A goal that is impossible to ever achieve at all in all development efforts

Answer: C

Q1081. What is the Chidamber and Kemerer (CK) metrics suite?

A. A collection of design patterns for modules and code
B. A comprehensive software testing tool suite and set
C. An established coding standard and guideline set only
D. A set of six object-oriented metrics: WMC, DIT, NOC, CBO, RFC, LCOM

Answer: D

Q1082. What is the Response For a Class (RFC) metric?

A. A network communication protocol metric and measure during the software lifecycle
B. A relational database performance metric and measure within the project scope
C. The count of all methods that can be invoked in response to a message to the class
D. A comprehensive software testing metric and measure for the development team

Answer: C

Q1083. What is the Lack of Cohesion of Methods (LCOM) metric?

A. A production deployment readiness metric and measure during implementation
B. A metric measuring how dissimilar methods are in a class based on the instance variables they use
C. A comprehensive software testing metric and measure in the system context
D. A measure of total source code line length and size throughout the project

Answer: B

Q1084. What is the Coupling Between Objects (CBO) metric?

A. A relational database join performance metric and score as a standard approach
B. A network bandwidth and latency metric and measurement in practice typically
C. A physical coupling measurement between parts and things for quality purposes
D. The count of classes to which a class is coupled through method calls, variables, or inheritance

Answer: D

Q1085. What is the concept of 'measurement dysfunction'?

A. Broken physical measurement tools and device parts by the organization
B. Errors in hardware measurement instruments and tools across all phases
C. When metrics are gamed or misused, leading to counterproductive behavior
D. Dysfunctional software that crashes frequently daily at every stage

Answer: C

Q1086. What is GQM+ Strategies?

A. A comprehensive software testing strategy and plan according to best practices
B. An extension of GQM that aligns measurement goals with organizational business goals
C. A recreational video gaming strategy guide and plan in a systematic way
D. A source code implementation coding strategy and plan for the project goals

Answer: B

Q1087. What is the concept of 'metric validation' in software engineering?

A. Validating application source code for errors and bugs by the development process
B. Testing software requirements for completeness and scope and its related activities
C. Ensuring that a metric actually measures what it claims to measure and is useful
D. Testing physical measurement tools for their accuracy within the system boundary

Answer: C

Q1088. What is the Number of Children (NOC) metric?

A. The total number of comments in the source code base
B. The total number of source code files in the project
C. The number of child processes currently running now
D. The count of immediate subclasses of a class in the inheritance tree

Answer: D

Q1089. What is the difference between size-oriented and function-oriented metrics?

A. Size-oriented uses LOC as the basis; function-oriented uses function points
B. Function-oriented metrics use LOC as the main basis as defined by standards
C. Size-oriented metrics use function points as basis for all project stakeholders
D. The two metric types are fundamentally identical overall over the entire lifecycle

Answer: A

Q1090. What is the concept of a 'balanced scorecard' in software metrics?

A. A scorecard for athletic sports competition events within the given constraints
B. A formal source code peer review evaluation and form by the project team members
C. A comprehensive software testing checklist form sheet in all development efforts
D. A strategic management framework measuring performance across multiple perspectives

Answer: D

Q1091. What is the concept of 'configuration identification'?

A. Identifying individual automated test case scenarios in a systematic way
B. Selecting and labeling configuration items and documenting their characteristics
C. Identifying specific software bugs and defect reports across all phases
D. Identifying individual team member profile records at every stage

Answer: B

Q1092. What is the difference between configuration management and change management?

A. CM only involves application source code files and code by the development process
B. Change management is always much broader in its scope within the system boundary
C. CM manages all configuration items; change management specifically handles the change request process
D. The two concepts are completely identical in all ways for the project goals according to best practices

Answer: C

Q1093. What is a Software Bill of Materials (SBOM)?

A. A project budget and financial document and report over the entire lifecycle
B. A grocery shopping list for the team members only and its related activities
C. A team member roster and directory listing sheet as defined by standards
D. A formal record of the components and dependencies in a piece of software

Answer: D

Q1094. What is Infrastructure as Code (IaC)?

A. Building physical infrastructure roads and buildings for all project stakeholders
B. A general-purpose coding language definition and spec within the given constraints
C. A comprehensive software testing framework and tool set in all development efforts
D. Managing and provisioning computing infrastructure through machine-readable configuration files

Answer: D

Q1095. What is the concept of 'immutable infrastructure'?

A. An approach where infrastructure components are replaced rather than modified, ensuring consistency
B. Permanent physical hardware installations and setups and related components
C. Infrastructure that physically cannot ever change form by the project team members
D. A specific type of database storage system and service to achieve project objectives

Answer: A

Q1096. What is semantic versioning?

A. Alphabetical versioning of all software release names for effective project outcomes
B. Using random numbering for all software version names in the engineering discipline
C. A versioning scheme using MAJOR.MINOR.PATCH to convey meaning about underlying changes
D. Date-based versioning of all software release versions as part of the methodology

Answer: C

Q1097. What is a monorepo strategy?

A. Multiple separate repos per each single project type
B. Storing multiple projects in a single version control repository
C. One separate repository per individual project file
D. Having absolutely no repository at all for the code

Answer: B

Q1098. What is the concept of 'configuration status accounting'?

A. Recording and reporting the status of configuration items and change requests throughout the project
B. Financial accounting and bookkeeping tasks and work in the development process
C. Tracking developer working hours per task and activity within the project scope
D. Counting total source code lines in the whole project during the software lifecycle

Answer: A

Q1099. What is a feature flag (feature toggle)?

A. A physical flag or banner for signaling and marking for the development team
B. A test indicator for pass or fail status result only in the system context
C. A technique allowing features to be enabled or disabled without deploying new code
D. A marker for identifying software bugs and issues only throughout the project

Answer: C

Q1100. What is the concept of 'GitOps'?

A. A comprehensive software testing approach and method only in practice typically as a standard approach
B. An operational framework where Git is the single source of truth for infrastructure and application deployment
C. An established coding standard and guideline set for teams by the organization at every stage
D. Only performing basic Git operations and nothing else during implementation for quality purposes

Answer: B

Q1101. What is the Expected Monetary Value (EMV) in risk analysis?

A. An employee salary expectation and forecast amount in practice typically
B. A project budget and financial metric and measurement as a standard approach
C. A product pricing strategy and approach and methodology by the organization
D. The product of probability and monetary impact of a risk, used in decision trees

Answer: D

Q1102. What is the concept of 'risk-driven development'?

A. Deliberately ignoring risks in the development process in a systematic way for the project goals
B. A comprehensive software testing approach and method only according to best practices
C. Developing intentionally risky software applications at every stage across all phases
D. A development approach that uses risk analysis to prioritize and guide architecture and design decisions

Answer: D

Q1103. What is a decision tree in risk management?

A. A test hierarchy and organization structure and setup and its related activities
B. A diagram showing decision choices, chance events, probabilities, and outcomes for risk analysis
C. A physical tree growing in a garden area outdoors within the system boundary
D. A source code tree and directory structure and layout by the development process

Answer: B

Q1104. What is sensitivity analysis in risk management?

A. An emotional sensitivity analysis technique approach over the entire lifecycle
B. A source code analysis technique and tool and method for all project stakeholders
C. A security vulnerability analysis technique and method as defined by standards
D. A technique that determines which risks have the greatest potential impact on project outcomes

Answer: D

Q1105. What is the concept of 'risk appetite' vs 'risk tolerance'?

A. The two concepts are completely identical in every way within the given constraints in all development efforts
B. Risk appetite is the level of risk an organization is willing to pursue; risk tolerance is the acceptable variation in outcomes
C. Risk appetite is always smaller than risk tolerance level by the project team members and related components
D. Risk tolerance always means absolutely zero risk accepted to achieve project objectives in the engineering discipline

Answer: B

Q1106. What is Monte Carlo simulation in risk management?

A. A technique using random sampling to simulate project outcomes and assess risk probability
B. A comprehensive software testing simulation and approach in the development process
C. A source code implementation coding simulation and test as part of the methodology
D. A recreational casino gambling game activity and event for effective project outcomes

Answer: A

Q1107. What is residual risk?

A. The risk that remains after risk responses have been implemented
B. A production deployment issue and concern and problem
C. No risk at all remaining after response is applied
D. A specific type of source code error found in the code

Answer: A

Q1108. What is secondary risk?

A. A risk that is considered less important than others during the software lifecycle
B. A specific type of reported software bug and defect for the development team
C. A new risk that arises as a direct result of implementing a risk response
D. A backup risk for contingency planning purposes only within the project scope

Answer: C

Q1109. What is the concept of 'risk-based testing'?

A. A testing approach that uses risk analysis to prioritize test cases and allocate testing effort
B. Testing software without a formal plan or schedule during implementation
C. Random and unstructured software testing approaches in the system context
D. Testing software with risky input data and scenarios throughout the project

Answer: A

Q1110. What is the Failure Mode and Effects Analysis (FMEA) in software?

A. A failure analysis report document summary and overview for quality purposes in practice typically
B. A systematic technique for identifying potential failure modes, their causes, and effects on the system
C. A production deployment analysis and report and summary by the organization at every stage
D. A source code implementation coding technique and method as a standard approach

Answer: B

Q1111. What is threat modeling?

A. Creating detailed threat assessment reports and summaries for the development team
B. A systematic approach to identifying, quantifying, and addressing security threats to a system
C. Modeling threats in three-dimensional space and models within the project scope
D. A specific type of software testing approach and method throughout the project

Answer: B

Q1112. What is the STRIDE threat model?

A. A source code implementation coding framework and library for quality purposes in practice typically
B. A comprehensive software testing model and approach only as a standard approach by the organization
C. A threat classification model: Spoofing, Tampering, Repudiation, Information Disclosure, DoS, Elevation of Privilege
D. A physical walking gait analysis technique and method in the system context during implementation

Answer: C

Q1113. What is the concept of 'security by design'?

A. Integrating security considerations into every phase of the software development lifecycle
B. Designing security logos and brand imagery and assets at every stage
C. Adding security features only after deployment to production across all phases
D. Writing security documentation exclusively and nothing else in a systematic way

Answer: A

Q1114. What is a zero-day vulnerability?

A. A vulnerability that is only one day old at most today within the system boundary
B. A previously unknown vulnerability that is exploited before a patch is available
C. A vulnerability with absolutely zero impact on the system according to best practices
D. A vulnerability that was fixed on day zero of discovery for the project goals

Answer: B

Q1115. What is the DREAD risk assessment model?

A. A comprehensive software testing model and approach only and its related activities
B. An established coding standard and guideline and rule set over the entire lifecycle
C. An assessment of fear and dread feelings and emotions by the development process
D. A model rating threats by Damage, Reproducibility, Exploitability, Affected users, Discoverability

Answer: D

Q1116. What is secure software development lifecycle (SSDLC)?

A. A comprehensive software testing lifecycle and approach for all project stakeholders
B. A production deployment lifecycle plan and schedule doc within the given constraints
C. An approach that integrates security practices into each phase of the traditional SDLC
D. A brand new programming language created from scratch as defined by standards

Answer: C

Q1117. What is the difference between symmetric and asymmetric encryption?

A. Asymmetric encryption uses only one single key for all by the project team members
B. The two encryption types are fundamentally the same thing in all development efforts
C. Symmetric encryption uses two different keys always now and related components
D. Symmetric uses the same key for encryption and decryption; asymmetric uses a key pair

Answer: D

Q1118. What is a Common Vulnerabilities and Exposures (CVE)?

A. A standardized system for identifying and naming publicly known cybersecurity vulnerabilities
B. A comprehensive software testing framework and tool suite for effective project outcomes
C. An established coding standard and guideline and rule set in the engineering discipline
D. A specific type of computer virus or malware and threat to achieve project objectives

Answer: A

Q1119. What is the concept of 'attack surface'?

A. The physical surface of a computer case and housing as part of the methodology
B. A network configuration and topology setup and structure during the software lifecycle
C. A specific type of user interface design and layout only in the development process
D. The total number of points where an unauthorized user can try to enter or extract data

Answer: D

Q1120. What is the principle of 'fail secure' vs 'fail open'?

A. The two concepts are fundamentally identical in all ways within the project scope
B. Fail secure allows all access on any failure or error event throughout the project
C. Fail open is always the more secure approach and method for the development team
D. Fail secure denies access by default on failure; fail open allows access on failure

Answer: D

Q1121. A company is experiencing frequent project delays and cost overruns. Which fundamental software engineering principle would most directly address this?

A. Hiring additional developers without changing methodology
B. Reducing documentation to accelerate coding activities
C. Adopting structured process models with clear milestones
D. Purchasing more powerful servers for the development team

Answer: C

Q1122. In Lehman's laws of software evolution, what does the law of continuing change state?

A. Software quality improves automatically without any effort
B. Software maintenance effort reduces with each new release
C. Software complexity decreases naturally over time periods
D. Software must continually adapt or become less satisfactory

Answer: D

Q1123. A team finds that fixing one bug consistently introduces two new defects. Which concept best explains this phenomenon?

A. Software entropy and increasing system complexity over time
B. Effective regression testing preventing all defect propagation
C. Proper documentation reducing overall maintenance difficulty
D. Successful modular design isolating component dependencies

Answer: A

Q1124. When applying the 'separation of concerns' principle, what is the primary architectural benefit achieved?

A. Documentation requirements are completely eliminated from work
B. Each module addresses a distinct feature independently of others
C. Testing becomes unnecessary due to simplified code structure
D. All system functionality is consolidated into a single module

Answer: B

Q1125. A startup must deliver a product quickly with uncertain requirements. Which combination of software engineering principles is most appropriate?

A. Comprehensive upfront documentation with rigid phase milestones
B. Iterative development with frequent stakeholder feedback cycles
C. Big bang integration testing after all modules are completed
D. Waterfall model with complete requirements before any coding

Answer: B

Q1126. What is Brooks' law and how does it impact late software projects?

A. Late projects benefit most from adding experienced engineers
B. Doubling the team size will exactly halve the delivery time
C. Adding people to a late project makes it even later overall
D. Project delays are always caused by insufficient team members

Answer: C

Q1127. In the context of sociotechnical systems, why must software engineers consider organizational factors?

A. Organizational factors have no measurable impact on software quality
B. Technical solutions must align with organizational processes and culture
C. Software operates independently of any human or organizational context
D. Engineering decisions should be based solely on technical merit alone

Answer: B

Q1128. A legacy system handles critical operations but uses obsolete technology. What is the recommended engineering approach?

A. Ignoring the system until it fails completely in production
B. Complete system rewrite without referencing existing functionality
C. Immediate shutdown followed by building from scratch entirely
D. Incremental modernization while maintaining system availability

Answer: D

Q1129. What distinguishes emergent system properties from designed properties in software engineering?

A. Emergent properties arise from component interactions not individual parts
B. Emergent properties can be fully predicted from individual components
C. Designed properties only appear after system deployment to production
D. Emergent properties are explicitly defined in requirements documentation

Answer: A

Q1130. How does Conway's law relate software architecture to organizational structure?

A. Technical architecture solely determines how teams are organized
B. Software architecture is always independent of team organization
C. Organizational hierarchy has no influence on system module design
D. System designs mirror the communication structure of the organization

Answer: D

Q1131. A government agency requires extensive documentation and formal sign-offs at each development stage. Which process model best fits this environment?

A. Waterfall model with formal reviews and documentation at each phase
B. Extreme programming with minimal documentation and rapid iterations
C. Scrum framework with two-week sprints and daily standup meetings
D. Kanban method with continuous flow and no predefined phase gates

Answer: A

Q1132. A team is building a safety-critical system for an aircraft. Why would the V-model be preferred over agile approaches?

A. V-model ensures rigorous verification and validation at each level
B. V-model eliminates the need for any form of requirements analysis
C. Agile approaches always produce higher quality safety-critical code
D. Agile methods provide more thorough documentation for regulators

Answer: A

Q1133. In the spiral model, how does the team determine the scope of each iteration?

A. By implementing all remaining features in every single spiral loop
B. By evaluating identified risks and prioritizing their mitigation efforts
C. By following a fixed predetermined schedule regardless of any risks
D. By randomly selecting features from the product backlog for building

Answer: B

Q1134. A project started with the waterfall model but requirements keep changing. What is the most pragmatic transition strategy?

A. Shift to an incremental model preserving completed phase artifacts
B. Continue with waterfall and reject all incoming change requests
C. Abandon process models entirely and use ad hoc development now
D. Restart the entire project from scratch using a different language

Answer: A

Q1135. What is the primary challenge of using evolutionary prototyping for large-scale enterprise systems?

A. Enterprise systems cannot use any iterative development technique
B. Large systems never benefit from any prototyping approach at all
C. Prototypes may evolve into poorly architected production systems
D. Evolutionary prototyping always produces perfectly structured code

Answer: C

Q1136. How does the Rational Unified Process (RUP) handle changing requirements compared to traditional plan-driven approaches?

A. RUP accommodates changes through iterative development within structured phases
B. RUP completely prevents any requirement changes after the inception phase ends
C. RUP requires a complete project restart whenever any requirement changes occur
D. RUP has no mechanism for handling requirements changes during development work

Answer: A

Q1137. A medical device company must comply with FDA validation requirements. How does this constraint affect process model selection?

A. It favors models with traceable verification and validation documentation
B. It mandates that no testing be performed until final system delivery
C. It eliminates all possible process models from consideration entirely
D. It requires using only agile methods without any formal documentation

Answer: A

Q1138. What is the fundamental trade-off between plan-driven and agile process models in terms of project governance?

A. Plan-driven offers more control and predictability; agile offers more flexibility and speed
B. Plan-driven and agile models provide identical levels of governance and predictability
C. Plan-driven models are always inferior to agile in every project context and scenario
D. Agile models provide more formal governance and control than plan-driven approaches do

Answer: A

Q1139. When applying the DevOps model, how does continuous integration change the traditional process model structure?

A. It separates development and operations into completely independent workflows
B. It requires all code to be written before any integration testing takes place
C. It merges development and operations phases into a continuous automated pipeline
D. It completely eliminates the need for any testing or quality assurance activities

Answer: C

Q1140. A startup pivots its product direction every three months based on market feedback. Which process model characteristic is most essential?

A. Fixed scope defined entirely at project start without any later changes
B. Long sequential phases with comprehensive documentation at every stage
C. Short iteration cycles with frequent delivery and customer feedback loops
D. Complete architecture designed before any implementation work begins

Answer: C

Q1141. A team's velocity has been declining over the past five sprints despite no personnel changes. What is the most likely root cause to investigate?

A. The team has become too skilled and experienced for the project work
B. Increasing technical debt making new development progressively harder
C. The product backlog has become too small with too few remaining items
D. Sprint duration has been too short for any meaningful work to occur

Answer: B

Q1142. How should a Scrum team handle a situation where the Product Owner frequently changes priorities mid-sprint?

A. Remove the Product Owner from the team to prevent further disruption to the process flow
B. Educate the Product Owner on sprint commitment and defer changes to next sprint planning
C. Immediately abandon current sprint work and start on the new priorities without question
D. Ignore all priority changes and continue working on the original plan without communication

Answer: B

Q1143. In SAFe (Scaled Agile Framework), what is the purpose of a Program Increment planning event?

A. To eliminate the need for individual team sprint planning sessions throughout the increment
B. To replace all Scrum ceremonies with a single comprehensive monthly planning meeting event
C. To align multiple agile teams on shared objectives and dependencies for a planning period
D. To assign specific tasks to individual developers across all teams in the entire organization

Answer: C

Q1144. A distributed agile team across three time zones is struggling with communication. Which practice would most effectively address this challenge?

A. Requiring all team members to work the same hours regardless of their local time zone
B. Establishing overlapping core hours and using asynchronous communication tools effectively
C. Eliminating all meetings and relying exclusively on email for all team communications
D. Splitting the team into independent groups that never need to communicate with each other

Answer: B

Q1145. What is the relationship between the Agile Manifesto values and the twelve agile principles?

A. The values replace the principles and make them unnecessary for any agile team practice
B. The values and principles are completely unrelated documents with different authors entirely
C. The principles contradict the values and should be followed instead of the manifesto values
D. The principles provide specific guidance for applying the four abstract manifesto values

Answer: D

Q1146. When transitioning from waterfall to agile, what is the most significant organizational challenge?

A. Shifting from command-and-control management to servant leadership and team autonomy
B. Purchasing new software development tools and IDE licenses for all team members now
C. Reducing the number of meetings from weekly status updates to no meetings at all
D. Renaming existing project roles to match agile terminology without changing practices

Answer: A

Q1147. How does technical debt impact agile teams differently than plan-driven teams?

A. Agile teams are immune to technical debt because they refactor code in every single sprint
B. Technical debt affects plan-driven teams more because they have longer development cycles
C. Agile teams feel the impact sooner due to shorter iteration cycles and continuous delivery
D. Plan-driven teams never accumulate technical debt due to their thorough upfront planning

Answer: C

Q1148. A team is practicing Scrum but the sprint goal is never met. The team completes individual stories but lacks cohesion. What is the underlying issue?

A. The sprint goal is not well-defined or the team is not collaborating toward a shared objective
B. The Product Owner is providing too few user stories for the team to work on each sprint
C. The sprint duration is too long and the team runs out of meaningful work before it ends
D. Individual team members are too skilled and complete work too quickly without coordination

Answer: A

Q1149. In Lean software development, what does 'eliminating waste' specifically refer to?

A. Removing activities that do not add value to the customer from the development process
B. Reducing the physical waste produced by the office including paper and plastic materials
C. Firing underperforming team members to improve the overall productivity of the team now
D. Deleting all source code comments and documentation to reduce the size of the codebase

Answer: A

Q1150. What is the difference between adaptive and predictive project management approaches in the context of agile?

A. Both approaches are identical in how they handle project scope and requirement changes fully
B. Adaptive approaches fix all requirements at start; predictive approaches change constantly now
C. Adaptive approaches embrace change and evolve plans; predictive approaches fix scope upfront
D. Predictive approaches are more responsive to change than adaptive approaches in all cases

Answer: C

Q1151. A healthcare system must comply with HIPAA regulations. How should these regulatory constraints be captured in requirements?

A. As project management constraints affecting only the delivery timeline and budget
B. As functional requirements describing the user interface layout and color scheme
C. As non-functional requirements specifying security, privacy, and audit trail standards
D. As optional features that can be implemented if time and budget allow during work

Answer: C

Q1152. During elicitation, stakeholders express contradictory requirements. What is the most effective resolution approach?

A. Implement all contradictory requirements simultaneously without resolving the conflicts
B. Facilitate negotiation sessions to find compromises and document agreed-upon priorities
C. Ignore the less senior stakeholder's requirements and implement only the manager's view
D. Postpone the entire project until all stakeholders naturally reach unanimous agreement

Answer: B

Q1153. What is the relationship between requirements volatility and project risk in software engineering?

A. Requirements volatility has no measurable impact on project risk or delivery outcomes
B. Requirements stability increases project risk because it prevents creative innovation work
C. Volatile requirements always reduce project risk by keeping the team adaptable and ready
D. Higher requirements volatility increases project risk through scope creep and rework costs

Answer: D

Q1154. How should safety-critical requirements be specified differently from standard functional requirements?

A. Using informal natural language descriptions without any additional precision or detail
B. With formal notation, rigorous verification criteria, and detailed failure mode analysis
C. Exactly the same way as any other functional requirement without special treatment given
D. By omitting them from documentation and relying on developer intuition during coding

Answer: B

Q1155. A team discovers during implementation that a key requirement is technically infeasible. What is the appropriate requirements engineering response?

A. Conduct impact analysis, negotiate alternatives with stakeholders, and update the SRS document
B. Cancel the entire project because a single requirement cannot be implemented as specified
C. Continue attempting to implement the infeasible requirement regardless of technical limitations
D. Silently remove the requirement from the SRS without informing any stakeholders about it

Answer: A

Q1156. What is the 'requirements gap' and how does it affect software projects?

A. It is the physical distance between the development team and the client office locations
B. It is the difference between documented requirements and actual stakeholder needs and expectations
C. It is the time delay between writing requirements and implementing them in source code only
D. It is the salary difference between requirements analysts and software development engineers

Answer: B

Q1157. How do goal-oriented requirements engineering approaches differ from traditional approaches?

A. They focus exclusively on technical implementation details rather than business-level objectives
B. They eliminate the need for any stakeholder communication during the requirements phase entirely
C. They focus on stakeholder goals and derive requirements from high-level objectives systematically
D. They produce identical results to traditional approaches but use different terminology and tools

Answer: C

Q1158. In a large enterprise system with hundreds of requirements, how should traceability be managed effectively?

A. By assigning one developer to memorize all requirement-to-code mappings without any tools
B. By avoiding traceability entirely since it adds overhead without providing any project value
C. By manually maintaining paper documents that cross-reference every requirement to code lines
D. Using automated traceability tools with bidirectional links between requirements and artifacts

Answer: D

Q1159. What is the impact of cultural differences on requirements elicitation in global software development projects?

A. Cultural differences affect communication styles, expectations, and interpretation of requirements
B. Cultural differences have no impact on requirements because software is a universal language
C. All cultures have identical approaches to expressing needs and defining system requirements
D. Cultural factors only affect project management and have no relevance to requirements work

Answer: A

Q1160. How does viewpoint-oriented requirements engineering handle the challenge of multiple perspectives?

A. It combines all viewpoints into one without distinguishing their sources or resolving conflicts
B. It selects a single viewpoint and ignores all other perspectives throughout the entire project
C. It systematically captures and reconciles requirements from different stakeholder viewpoints
D. It eliminates the need for stakeholder involvement by relying solely on developer assumptions

Answer: C

Q1161. A project is showing a Cost Performance Index (CPI) of 0.8 and Schedule Performance Index (SPI) of 0.9. What does this indicate?

A. The project is under budget by 20% and ahead of schedule by 10% of planned work
B. The project is over budget by 20% and behind schedule by 10% of planned progress
C. The project is exactly on budget and on schedule according to the baseline plan
D. The project has been completed successfully with all deliverables meeting standards

Answer: B

Q1162. How should a project manager handle a situation where the estimated effort exceeds the available budget?

A. Cancel the project immediately without exploring any alternative approaches or solutions
B. Proceed with the full scope and hope the team can work faster than the estimates suggest
C. Add more developers to reduce the schedule without adjusting the budget or scope at all
D. Negotiate scope reduction with stakeholders while maintaining core functionality intact

Answer: D

Q1163. In Brooks' Mythical Man-Month, why does adding developers to a late project make it later?

A. New developers instantly become productive and reduce the workload on existing team members
B. Projects always finish on time regardless of team size changes during the development period
C. Adding developers has no measurable effect on project schedule in any known situation today
D. Communication overhead grows quadratically and new members need ramp-up training time

Answer: D

Q1164. A team uses PERT estimation with optimistic=2 weeks, most likely=4 weeks, and pessimistic=12 weeks. What is the expected duration?

A. Five weeks based on the weighted average formula used in PERT estimation technique
B. Two weeks because PERT always selects the most optimistic estimate for planning work
C. Twelve weeks because PERT always uses the pessimistic estimate for safety margin only
D. Four weeks because the most likely estimate is always the final expected duration value

Answer: A

Q1165. How does the theory of constraints apply to software project management?

A. It requires constraining all team members to work at exactly the same speed and pace
B. It identifies and focuses improvement efforts on the bottleneck limiting overall throughput
C. It suggests adding constraints to every activity to improve project discipline overall
D. It eliminates all project constraints by removing deadlines and budget limitations now

Answer: B

Q1166. A project manager discovers that two critical team members plan to leave mid-project. What proactive measures should be taken?

A. Ignore the risk and assume the remaining team can absorb all additional workload easily
B. Immediately cancel the project since replacing team members is impossible to accomplish
C. Reduce product quality standards so the remaining team can deliver with less total effort
D. Document knowledge, cross-train team members, and plan for replacement hiring immediately

Answer: D

Q1167. What is the relationship between project size and defect density in software engineering?

A. Project size has absolutely no correlation with defect density in any published research data
B. Defect density remains perfectly constant regardless of project size or complexity levels
C. Smaller projects always have more defects than larger projects due to less thorough testing
D. Larger projects tend to have higher defect density due to increased complexity and coupling

Answer: D

Q1168. How should a project manager balance the competing demands of multiple stakeholders with different priorities?

A. Randomly select which stakeholder requirements to implement without any systematic approach
B. Ignore all stakeholder input and make decisions based solely on the development team opinion
C. Use stakeholder analysis to understand influence and interest, then negotiate compromises
D. Always prioritize the requirements of the most senior stakeholder without any negotiation

Answer: C

Q1169. What is Monte Carlo simulation used for in project management and how does it improve estimation accuracy?

A. It eliminates all uncertainty from project estimates by using a fixed mathematical formula only
B. It provides a single deterministic estimate that is always accurate for every project situation
C. It replaces the need for expert judgment by using random numbers without any project data
D. It runs thousands of simulations with varying inputs to produce probability-based schedule forecasts

Answer: D

Q1170. In agile project management, how does the concept of 'servant leadership' change the traditional project manager role?

A. The manager delegates all responsibility and has no involvement in the project activities at all
B. The manager increases direct control over all individual tasks and decisions made by the team
C. The manager role is completely eliminated and no leadership exists within agile project teams
D. The manager focuses on removing impediments and empowering the team rather than directing work

Answer: D

Q1171. A system needs to support multiple database backends (MySQL, PostgreSQL, MongoDB) without changing business logic. Which design approach is most appropriate?

A. Hardcoding database queries directly into business logic classes for maximum performance
B. Abstract Factory pattern with database-specific factory implementations for each backend
C. Using global variables to store database connection strings throughout the application
D. Writing separate copies of the business logic for each database backend independently

Answer: B

Q1172. How does the SOLID principles violation manifest in a God class anti-pattern?

A. The class violates SRP by handling too many responsibilities and having many reasons to change
B. The class follows SRP perfectly by focusing on a single well-defined responsibility only
C. The class demonstrates ideal OCP by being completely closed to any future extensions work
D. The class exemplifies LSP by being perfectly substitutable for all of its subtype classes

Answer: A

Q1173. A real-time trading system must process millions of events per second. How should the design balance performance with maintainability?

A. Sacrifice all code readability and maintainability to achieve maximum raw processing speed
B. Implement all processing in a single monolithic function for maximum execution throughput
C. Use the simplest possible design without considering performance requirements at any point
D. Use event-driven architecture with optimized hot paths while keeping non-critical paths clean

Answer: D

Q1174. What is the relationship between design patterns and anti-patterns in software engineering?

A. Anti-patterns are always better than design patterns for solving complex engineering problems
B. Design patterns cause problems while anti-patterns provide effective solutions for all cases
C. Design patterns solve recurring problems; anti-patterns are commonly used but ineffective solutions
D. Design patterns and anti-patterns are identical concepts with interchangeable names and usage

Answer: C

Q1175. How does the Hexagonal Architecture (Ports and Adapters) improve testability of business logic?

A. It isolates business logic from external dependencies through ports allowing easy mock injection
B. It embeds external dependencies directly into business logic making testing straightforward now
C. It makes testing harder by creating multiple layers of indirection between all components now
D. It eliminates the need for any testing by ensuring the business logic is always correct already

Answer: A

Q1176. A development team is debating between using inheritance hierarchies and composition for code reuse. What factors should guide this decision?

A. Always use composition because inheritance should never be used in any software system today
B. Use inheritance for genuine is-a relationships; prefer composition for flexible behavior reuse
C. Always use inheritance because it is fundamentally superior to composition in every scenario
D. The choice is purely aesthetic and has no impact on system flexibility or maintainability

Answer: B

Q1177. How does the Command pattern enable undo functionality in software applications?

A. The Command pattern only supports redo operations and cannot implement undo functionality now
B. Undo is implemented by restarting the entire application from its initial state each time needed
C. The Command pattern prevents any actions from being undone once they have been executed fully
D. Each command object encapsulates an action and its inverse allowing execution reversal on demand

Answer: D

Q1178. What is the impact of circular dependencies on software design quality and maintainability?

A. They simplify the build process by reducing the number of modules that need compilation work
B. They have no measurable impact on software design quality or long-term maintainability at all
C. They increase coupling, complicate builds, and make modules impossible to test in isolation easily
D. They improve design quality by ensuring all modules are tightly integrated with each other now

Answer: C

Q1179. In domain-driven design, how does the Bounded Context pattern help manage complexity in large systems?

A. It requires a single unified domain model used identically across all parts of the entire system
B. It eliminates the need for any domain modeling by using generic data structures everywhere now
C. It defines clear boundaries where specific domain models apply preventing model conflicts globally
D. It increases complexity by requiring every team to understand all domain models in the system

Answer: C

Q1180. How should a designer handle the tension between the DRY principle and the need for module independence?

A. Accept controlled duplication when shared abstractions would create inappropriate coupling between modules
B. Always eliminate every instance of duplication regardless of the coupling it introduces between parts
C. Force all modules to share a single implementation even when their requirements diverge significantly
D. Ignore the DRY principle entirely because code duplication has no negative consequences on quality

Answer: A

Q1181. A company needs to migrate a monolithic e-commerce platform to microservices. What is the recommended architectural migration strategy?

A. Identify bounded contexts, extract services incrementally, and use an API gateway for routing
B. Rewrite the entire system from scratch in a new language without referencing existing code
C. Keep the monolith exactly as is and simply rename it as a microservices architecture now
D. Split the monolith randomly into services without analyzing domain boundaries or dependencies

Answer: A

Q1182. How does the CAP theorem constrain the design of distributed database architectures?

A. The CAP theorem only applies to relational databases and not to any NoSQL database systems
B. A distributed system can guarantee at most two of consistency, availability, and partition tolerance
C. Partition tolerance is optional in distributed systems and can be safely ignored in all designs
D. A distributed system can easily achieve all three properties simultaneously without any trade-offs

Answer: B

Q1183. A streaming analytics platform must process ten million events per second with sub-second latency. Which architectural pattern is most suitable?

A. Single-threaded monolithic application processing events sequentially one at a time in order
B. Batch processing architecture that collects events for hourly processing in large bulk operations
C. Traditional request-response architecture with synchronous database queries for each single event
D. Event sourcing with partitioned stream processing and in-memory computation for hot data paths

Answer: D

Q1184. What are the architectural implications of choosing eventual consistency over strong consistency?

A. Eventual consistency has no implications and behaves identically to strong consistency in all cases
B. Strong consistency always provides better performance than eventual consistency in every scenario
C. Eventual consistency eliminates all data inconsistency problems permanently without any trade-offs
D. The system gains availability and performance but must handle temporary data inconsistencies gracefully

Answer: D

Q1185. How should an architect design a system to be resilient against cascading failures in a microservices environment?

A. Remove all error handling to reduce code complexity and improve overall system performance metrics
B. Implement circuit breakers, bulkheads, and graceful degradation patterns between all service boundaries
C. Use a single shared database for all services to eliminate any communication failures between them
D. Make every service synchronously dependent on every other service to ensure data consistency always

Answer: B

Q1186. What is the architectural significance of Conway's law when designing large-scale distributed systems?

A. Conway's law has been disproven and has no relevance to modern software architecture decisions
B. System architecture should be designed without any consideration of team organization structure
C. Team structure should align with desired system architecture to avoid communication mismatches
D. Conway's law only applies to monolithic systems and is irrelevant for distributed architectures

Answer: C

Q1187. A healthcare system requires 99.999% availability. How does this requirement influence the architectural decisions?

A. It requires a single centralized server to minimize complexity and potential points of failure in system
B. It mandates redundancy, automatic failover, geographic distribution, and elimination of single points of failure
C. It means the system should avoid any redundancy to reduce the number of components that can fail
D. It has no impact on architecture because availability is purely an operational and infrastructure concern

Answer: B

Q1188. How does the saga pattern handle distributed transactions across microservices?

A. It uses traditional two-phase commit protocol locking all services until the transaction completes fully
B. It coordinates a sequence of local transactions with compensating actions for failure rollback scenarios
C. It requires all microservices to share a single database to ensure transactional consistency always
D. It avoids any form of transaction management and accepts data corruption as an acceptable outcome

Answer: B

Q1189. What architectural considerations are necessary when designing a multi-tenant SaaS application?

A. Multi-tenant applications should use a separate deployed instance for every individual tenant always
B. All tenants must share identical configurations with no customization options available for any tenant
C. Data isolation strategies, per-tenant customization, scalability, and fair resource allocation mechanisms
D. Multi-tenancy requires no special architectural considerations beyond a standard single-user application

Answer: C

Q1190. How does the sidecar pattern enhance service mesh architectures in cloud-native applications?

A. It deploys supporting functionality alongside each service without modifying the service code itself
B. It embeds all infrastructure concerns directly into business logic code of every service instance
C. It requires every service to implement its own service discovery and load balancing logic fully
D. It eliminates the need for any networking between services by running everything in one process

Answer: A

Q1191. A healthcare application is used by elderly patients with limited technical experience. Which UI design principles should be prioritized?

A. Large touch targets, high contrast, simple navigation, clear typography, and minimal cognitive load
B. Advanced keyboard shortcuts, minimal visual feedback, and hidden navigation behind icon-only menus
C. Small fonts, complex multi-level menus, gesture-only navigation, and low contrast color schemes
D. Dense information displays, small buttons, animation-heavy interfaces, and technical terminology use

Answer: A

Q1192. How does cognitive load theory influence the number of options presented in a UI at one time?

A. The number of options has no measurable impact on user performance or satisfaction in any context
B. Cognitive load theory suggests options should never be limited because users can process infinite items
C. Too many options increase cognitive load and decision time, following Hick's law principles closely
D. Users always prefer having maximum options displayed simultaneously regardless of task complexity

Answer: C

Q1193. A team is designing a dashboard for data analysts who monitor real-time metrics. What design pattern best supports their workflow?

A. A slideshow that automatically rotates through metrics without any user control or interaction
B. Full-screen modal dialogs for each individual metric requiring sequential manual navigation
C. A single fixed text-based report with no interactive elements or visualization capabilities
D. Overview plus detail pattern with drill-down capabilities and customizable widget arrangements

Answer: D

Q1194. How should error messages be designed to minimize user frustration and support error recovery?

A. Cryptic error codes without any explanation or guidance for users on how to resolve the problem
B. Blaming the user for causing the error and demanding they contact technical support immediately
C. Clear language explaining what went wrong, why it happened, and specific steps to fix the issue
D. Hiding all errors silently without any notification so the user does not experience any frustration

Answer: C

Q1195. What is the impact of animation and micro-interactions on perceived application performance?

A. All animations slow down perceived performance and should be completely eliminated from every app
B. Animations should only be used for decorative purposes and never for functional feedback to users
C. Animations have no measurable impact on how users perceive the speed of the application interface
D. Well-designed animations mask loading times and provide continuity making the app feel faster overall

Answer: D

Q1196. A multi-language application serves users across twenty countries. What UI design challenges must be addressed?

A. Text expansion and contraction, RTL layout support, cultural color meanings, and locale-specific formats
B. Internationalization only requires translating text strings and has no impact on layout or design work
C. All languages and cultures use identical UI conventions so no design adaptations are ever necessary
D. Multi-language support only affects backend code and has no visible impact on the user interface

Answer: A

Q1197. How does the concept of 'dark patterns' in UI design undermine user trust?

A. Dark patterns refer exclusively to using dark color themes and have no ethical implications at all
B. Dark patterns manipulate users into unintended actions through deceptive design techniques intentionally
C. Dark patterns are always beneficial because they guide users toward the best possible decisions
D. Dark patterns have no impact on user trust because most users cannot detect manipulative design

Answer: B

Q1198. A financial trading application must display rapidly changing data without overwhelming the user. How should the UI handle this?

A. Use identical visual treatment for all data regardless of importance or rate of change observed
B. Use visual prioritization, controlled update frequencies, and highlight only significant data changes
C. Remove all real-time data and only show daily summary reports to reduce information overload
D. Display every data change immediately with full-screen alerts for each individual price movement

Answer: B

Q1199. What is the relationship between information architecture and UI design in creating effective digital products?

A. Information architecture defines content structure and organization that UI design then visualizes
B. UI design determines information architecture since visual layout always precedes content structure
C. Information architecture only applies to physical spaces and has no relevance to digital products
D. Information architecture and UI design are completely unrelated disciplines with no overlap at all

Answer: A

Q1200. How should a design team approach creating an inclusive design that works for users with motor, visual, cognitive, and auditory disabilities?

A. Design only for the average user and provide a separate simplified version for disabled users later
B. Assistive technologies handle all accessibility needs so designers do not need to consider them now
C. Follow WCAG guidelines, test with assistive technologies, and involve users with disabilities in testing
D. Accessibility is a legal requirement only and does not require any changes to the actual design work

Answer: C

Q1201. A team is experiencing frequent merge conflicts in a large codebase. What implementation practices would most effectively reduce this problem?

A. Larger infrequent commits combining many changes to reduce the total number of merge operations
B. Smaller frequent commits, feature branching strategies, and modular code with clear ownership areas
C. Having all developers work on the same branch and file simultaneously without any coordination
D. Avoiding version control entirely and using shared network drives for all source code storage

Answer: B

Q1202. How does test-driven development (TDD) change the implementation workflow compared to traditional development?

A. Tests are completely eliminated from the development process to accelerate feature delivery speed
B. TDD and traditional development follow identical workflows with no differences in approach at all
C. Implementation is completed fully before any tests are considered or written for the codebase
D. Tests are written before implementation code, driving design decisions and ensuring test coverage

Answer: D

Q1203. A legacy codebase has no automated tests. What is the safest approach to introduce testing?

A. Delete all existing code and replace it with test stubs that will be implemented later on
B. Skip testing entirely since the codebase has worked without tests for its entire lifetime so far
C. Write characterization tests capturing current behavior before making any code changes to system
D. Rewrite the entire codebase from scratch with tests rather than adding tests to existing code

Answer: C

Q1204. How does feature flag implementation enable continuous deployment while managing risk?

A. Feature flags completely prevent any bugs from reaching the production environment automatically
B. Feature flags are only useful for UI changes and cannot be applied to backend logic or services
C. Flags allow deploying code to production with features disabled, enabling gradual controlled rollout
D. Flags eliminate the need for any testing since features can be toggled off if problems occur now

Answer: C

Q1205. What is the impact of choosing a monorepo versus polyrepo strategy on implementation workflow?

A. Monorepos are only suitable for small projects and cannot scale to large enterprise codebases
B. Monorepos and polyrepos have identical workflow implications with no meaningful differences at all
C. Monorepos simplify cross-project changes but require sophisticated build and CI tooling at scale
D. Polyrepos are always superior because they eliminate all cross-project dependency management needs

Answer: C

Q1206. A performance-critical system component is a bottleneck. How should the team approach optimization?

A. Apply every known optimization technique to all code regardless of whether it is a bottleneck
B. Rewrite the entire codebase in assembly language without profiling or measuring performance first
C. Profile to identify actual bottlenecks, optimize measured hot paths, and benchmark improvements
D. Avoid any optimization since premature optimization is the root of all evil in every situation

Answer: C

Q1207. How does clean code philosophy address the trade-off between code readability and cleverness?

A. It treats readability and cleverness as identical qualities that never conflict with each other
B. It suggests that code readability is irrelevant since compilers handle all code interpretation
C. It encourages maximally clever code because shorter code is always better regardless of clarity
D. It prioritizes readability over cleverness because code is read far more often than it is written

Answer: D

Q1208. What are the implementation challenges of handling distributed system failures gracefully?

A. All distributed failures can be prevented entirely through proper implementation techniques alone
B. Distributed systems never experience failures because modern networks are perfectly reliable now
C. Network partitions, partial failures, and timeout handling require careful idempotent retry design
D. Failure handling is identical in distributed and single-machine systems with no added complexity

Answer: C

Q1209. How does immutable infrastructure change the deployment and implementation practices?

A. Immutable infrastructure prevents any updates or patches from being applied to the system ever
B. Immutable infrastructure is identical to traditional mutable server management in all aspects
C. Servers are continuously patched and modified in place to apply updates and configuration changes
D. Servers are never modified after creation; changes require building and deploying new instances

Answer: D

Q1210. What is the role of code complexity metrics like cyclomatic complexity in guiding implementation decisions?

A. Complexity metrics should only be used for documentation purposes and never influence coding work
B. Higher complexity always means better code because it handles more edge cases and scenarios now
C. Cyclomatic complexity has no correlation with defect rates or maintenance difficulty in any study
D. High complexity indicates code that is harder to test, maintain, and more likely to contain defects

Answer: D

Q1211. A team has 90% code coverage but still experiences production bugs. What does this reveal about their testing strategy?

A. High coverage does not guarantee test quality; tests may lack meaningful assertions and edge cases
B. Code coverage is meaningless and should be completely ignored as a testing quality metric now
C. The remaining ten percent of uncovered code contains all bugs and coverage should reach exactly full
D. Ninety percent coverage guarantees that ninety percent of all possible bugs have been detected

Answer: A

Q1212. How does property-based testing complement example-based testing?

A. It generates hundreds of random inputs to test properties that should hold for all valid inputs
B. It is identical to example-based testing and provides no additional value to the test suite now
C. It only tests a single predefined example and is less thorough than example-based approaches
D. It replaces example-based testing entirely and provides complete coverage of all possibilities

Answer: A

Q1213. A safety-critical aviation system requires DO-178C compliance. How does this affect the testing approach?

A. It allows the team to choose any testing approach without following any specific testing standards
B. It only requires basic unit testing without any structural coverage or formal verification work
C. It mandates structural coverage analysis, requirements-based testing, and formal verification methods
D. It eliminates the need for any testing because certification replaces all quality activities now

Answer: C

Q1214. What is the oracle problem in software testing and how does it affect test automation?

A. The oracle problem has been completely solved and no longer affects any testing activities today
B. The oracle problem only applies to manual testing and has no impact on automated testing work
C. Difficulty in determining expected outputs for complex scenarios limits automated test effectiveness
D. It refers to the challenge of selecting which database system to use for storing test data only

Answer: C

Q1215. How does chaos engineering differ from traditional testing approaches?

A. It replaces all traditional testing and eliminates the need for unit and integration test suites
B. It intentionally introduces failures in production to verify system resilience and recovery capability
C. It follows traditional scripted test cases executed in a controlled staging environment only today
D. It avoids any deliberate failure injection and relies on monitoring to detect issues passively

Answer: B

Q1216. A microservices architecture has complex service interactions. What testing strategy best addresses integration risks?

A. Full end-to-end testing of every possible service interaction path for complete coverage goal
B. Contract testing verifying service interfaces plus consumer-driven contracts between service pairs
C. Skipping all testing and relying on production monitoring to detect integration failures only
D. Only unit testing individual services without any integration or contract testing between them

Answer: B

Q1217. What is the relationship between testability and software design quality?

A. Poorly designed code is always easier to test than well-designed code due to simpler structure
B. Highly testable code typically exhibits good design qualities like loose coupling and high cohesion
C. Testability has no relationship with design quality and they are completely independent concerns
D. Testability should never influence design decisions because it compromises performance always

Answer: B

Q1218. How should a team prioritize test automation efforts when resources are limited?

A. Automate high-value repetitive tests first: smoke tests, regression suites, and critical path scenarios
B. Automate only the tests that are easiest to write regardless of their value to the project now
C. Avoid all automation and rely entirely on manual testing regardless of the available resources
D. Automate random tests without considering their value or frequency of execution by the team

Answer: A

Q1219. What is the impact of flaky tests on development team productivity and confidence?

A. Flaky tests are beneficial because they simulate real-world conditions and improve test coverage
B. Flaky tests improve team productivity by adding unpredictability that keeps developers alert now
C. Flaky tests erode confidence in the test suite and cause teams to ignore legitimate test failures
D. Flaky tests have no measurable impact on team productivity or confidence in the testing suite

Answer: C

Q1220. How does shift-left testing change the economics of defect detection in software projects?

A. Testing earlier increases costs because more comprehensive testing infrastructure is needed upfront
B. Finding defects earlier in development is significantly cheaper than finding them in production later
C. Shift-left testing has no impact on the cost of defect detection at any stage of development
D. Defects found in production are always cheaper to fix than those found during early development

Answer: B

Q1221. A legacy system with no documentation or original developers must be maintained. What is the recommended approach?

A. Make changes randomly and deploy them immediately without understanding the system structure
B. Use reverse engineering and program comprehension techniques to recover design knowledge first
C. Refuse to maintain the system and recommend shutting it down immediately without replacement
D. Rewrite the entire system from scratch without studying the existing system behavior at all

Answer: B

Q1222. How does Lehman's law of conservation of organizational stability affect maintenance planning?

A. Adding more developers always proportionally increases the rate of maintenance activity and output
B. The rate of development activity tends to remain constant over a system's lifetime regardless of resources
C. Organizational stability has no measurable effect on maintenance planning or resource allocation
D. Maintenance effort always decreases over time as the system becomes more stable and mature now

Answer: B

Q1223. A system experiences performance degradation every few months requiring manual restarts. What maintenance strategy addresses this systematically?

A. Add more hardware resources each time degradation occurs without investigating the root cause
B. Ignore the degradation until the system completely fails and then perform emergency maintenance
C. Manually restart the system only when users complain about noticeable performance problems now
D. Implement proactive software rejuvenation with automated scheduled restarts and resource cleanup

Answer: D

Q1224. What is the relationship between software architecture erosion and increasing maintenance costs?

A. Architecture erosion increases coupling and decreases cohesion making changes progressively harder
B. Architecture erosion has no impact on maintenance costs because architecture is only relevant initially
C. Erosion decreases maintenance costs by simplifying the system structure through organic evolution now
D. Architecture quality always improves over time through maintenance activities without deliberate effort

Answer: A

Q1225. How should a maintenance team prioritize between corrective, adaptive, perfective, and preventive maintenance requests?

A. Preventive maintenance should always have lowest priority since it addresses only future problems
B. Prioritize critical corrections first, then assess adaptive, perfective, and preventive by business value
C. All maintenance types should be treated with equal priority without any systematic prioritization
D. Always complete all corrective maintenance before considering any other type of maintenance request

Answer: B

Q1226. What is the economic justification for investing in preventive maintenance of software systems?

A. Preventing defects is significantly cheaper than fixing them after they manifest in production use
B. There is no economic basis for preventive maintenance since all software eventually needs replacing
C. Preventive maintenance never provides any return on investment and should always be avoided now
D. The cost of preventive maintenance always exceeds the cost of fixing defects after they occur

Answer: A

Q1227. How does microservices architecture change the maintenance approach compared to monolithic systems?

A. Individual services can be maintained, deployed, and scaled independently reducing maintenance scope
B. Microservices and monolithic systems require identical maintenance approaches without differences
C. Microservices eliminate all maintenance needs because services are small enough to rewrite quickly
D. Monolithic systems are always easier to maintain than microservices in every possible scenario now

Answer: A

Q1228. A company wants to decide whether to maintain or replace a legacy system. What factors should drive this decision?

A. Business value, maintenance cost trajectory, technical risk, and availability of replacement options
B. Legacy systems should always be maintained indefinitely regardless of costs or technical risks
C. Only the age of the system should be considered since older systems should always be replaced now
D. The decision should be based solely on the programming language used regardless of other factors

Answer: A

Q1229. What is the difference between software maintenance and software evolution from a theoretical perspective?

A. Maintenance implies keeping software functional; evolution implies continuous adaptation and growth
B. Maintenance and evolution are identical concepts with no theoretical distinction between them now
C. Maintenance is a proactive process while evolution is purely reactive to environmental changes
D. Evolution only occurs during initial development while maintenance happens exclusively after release

Answer: A

Q1230. How do maintenance metrics like Mean Time to Repair (MTTR) guide maintenance process improvement?

A. MTTR measures repair speed, identifying bottlenecks in diagnosis, implementation, and deployment phases
B. MTTR has no practical value for improving maintenance processes and should not be tracked at all
C. Lower MTTR values always indicate worse maintenance performance and should be increased over time
D. MTTR only measures the time to find bugs and does not include the time to fix and deploy them now

Answer: A

Q1231. How does the Deming cycle (PDCA) apply to continuous quality improvement in software?

A. PDCA requires completing all four steps simultaneously without any sequential ordering of them
B. The cycle should only be executed once and never repeated for continuous improvement purposes
C. The Deming cycle only applies to manufacturing and has no relevance to software quality at all
D. Plan improvements, Do implement them, Check results, Act to standardize or adjust the approach

Answer: D

Q1232. A company at CMMI Level 2 wants to reach Level 3. What fundamental change is required?

A. Reducing the number of quality activities to simplify the overall process framework at the org
B. Transitioning from project-level managed processes to organization-wide defined standard processes
C. Maintaining the same project-level processes without any standardization across the organization
D. Only purchasing more expensive quality assurance tools without changing any existing processes now

Answer: B

Q1233. What is the relationship between technical debt and software quality degradation over time?

A. Technical debt accumulates quality compromises that progressively degrade system quality over time
B. Accumulating technical debt always improves software quality by simplifying the system structure
C. Technical debt only affects development speed and never impacts the quality of the software now
D. Technical debt has no relationship with software quality and can be safely ignored by all teams

Answer: A

Q1234. How should an organization measure the return on investment of quality assurance activities?

A. Calculate ROI based solely on the number of QA engineers hired regardless of their actual output
B. Only measure the number of defects found without considering the cost of finding or preventing them
C. ROI of QA cannot be measured because quality improvements are intangible and subjective in all cases
D. Compare prevention and appraisal costs against reduction in failure costs and customer satisfaction gains

Answer: D

Q1235. A software product consistently fails in production despite passing all quality gates. What systemic issue might explain this?

A. Quality gates always guarantee production quality and any failures must be caused by hardware now
B. The quality gates are too strict and should be relaxed to allow more defects through to production
C. Quality gates may not adequately represent real-world conditions and usage patterns of the system
D. The development team is too skilled and the quality gates cannot keep up with their code changes

Answer: C

Q1236. How do organizational culture and management commitment affect quality assurance effectiveness?

A. Management commitment has no impact on QA because quality is solely the responsibility of testers
B. Strong management commitment and quality culture are essential for effective QA adoption and practice
C. Excessive management involvement always reduces QA effectiveness by creating bureaucratic overhead
D. Organizational culture is irrelevant to quality outcomes since only tools and processes matter now

Answer: B

Q1237. What is the Six Sigma approach and how does it apply to software quality management?

A. It aims to reduce process variation to achieve fewer than 3.4 defects per million opportunities made
B. Six Sigma only works for hardware manufacturing and has no proven application in software at all
C. It requires eliminating all defects completely with zero tolerance for any imperfection in product
D. Six Sigma increases process variation intentionally to improve creativity in software development

Answer: A

Q1238. How does model-based testing support quality assurance for complex software systems?

A. It eliminates the need for any testing by mathematically proving software correctness from models
B. It replaces quality assurance entirely by generating defect-free code directly from the models
C. It only works for simple systems and cannot handle the complexity of real-world applications now
D. It generates test cases from system models ensuring systematic coverage of modeled behaviors fully

Answer: D

Q1239. What is the impact of shifting quality responsibility to the entire development team rather than a separate QA group?

A. It doubles the quality effort because both developers and QA testers do identical redundant work
B. It has no impact on quality outcomes because only dedicated QA teams can effectively ensure quality
C. It integrates quality thinking into all activities leading to earlier defect detection and prevention
D. It eliminates quality entirely because developers do not have the skills to perform QA activities

Answer: C

Q1240. How should quality requirements be quantified and measured for a performance-critical real-time system?

A. Quality requirements for real-time systems cannot be quantified and must remain purely qualitative
B. Only measure response time and ignore all other quality attributes for real-time system evaluation
C. Define measurable SLAs for latency, throughput, and availability with continuous monitoring infrastructure
D. Use the same quality metrics as batch processing systems without any adaptation for real-time needs

Answer: C

Q1241. A team is using LOC as the primary productivity metric. What are the fundamental problems with this approach?

A. LOC is technology-independent and measures delivered value perfectly for all project contexts
B. LOC is the most accurate and comprehensive metric for measuring developer productivity always
C. LOC penalizes efficient code, varies by language, and does not measure delivered business value
D. There are no problems with using LOC as a productivity metric in any development environment

Answer: C

Q1242. How should metrics be used to avoid Goodhart's law in software development teams?

A. Set aggressive metric targets and penalize teams that fail to meet them to ensure compliance
B. Goodhart's law does not apply to software metrics and can be safely ignored by all managers
C. Use metrics for insight rather than targets because optimizing for metrics distorts the behavior
D. Use a single metric for all performance evaluations to ensure simplicity and consistency now

Answer: C

Q1243. A project shows decreasing defect density but increasing customer complaints. What does this discrepancy reveal?

A. Testing may miss requirement-level defects that affect user experience but not code-level quality
B. Customer complaints are irrelevant to software quality and should be ignored by the team always
C. The metrics system is working perfectly and no investigation is needed for this discrepancy at all
D. Decreasing defect density always means improved customer satisfaction without any exceptions now

Answer: A

Q1244. How does the Balanced Scorecard approach apply to software engineering measurement?

A. It measures performance across financial, customer, process, and learning dimensions holistically
B. It only measures financial metrics and ignores all other aspects of software project performance
C. It measures a single dimension of performance and provides a narrow view of project health only
D. It is exclusively a hardware manufacturing tool with no application to software engineering now

Answer: A

Q1245. What is the relationship between code churn metrics and defect prediction in software projects?

A. Code churn has no correlation with defect rates and provides no predictive value for quality now
B. Lower code churn always indicates more defects because unchanged code accumulates hidden bugs
C. Code churn only measures developer activity and has no relationship to code quality whatsoever
D. High code churn in specific files correlates with higher defect probability in those areas later

Answer: D

Q1246. How should an organization establish a metrics program without creating a dysfunctional measurement culture?

A. Publicly rank individual developers by all metrics and reward the highest performers exclusively
B. Use metrics exclusively for performance reviews and salary decisions without any other purpose
C. Focus on team-level process improvement metrics rather than individual performance punishment data
D. Collect as many metrics as possible regardless of their relevance to organizational goals now

Answer: C

Q1247. What is the role of benchmarking in software metrics and what are its limitations?

A. Benchmarking provides perfectly comparable results across all organizations without any limitations
B. It is impossible to benchmark software organizations because no two projects are ever similar now
C. Benchmarking compares performance against peers but context differences limit direct comparisons
D. Benchmarking only works for hardware measurements and has no application in software contexts

Answer: C

Q1248. How do object-oriented metrics like CK metrics suite differ from traditional complexity metrics?

A. CK metrics and traditional metrics measure identical properties with identical calculation methods
B. CK metrics only apply to procedural programming and cannot be used for object-oriented systems
C. Traditional metrics are always superior to CK metrics for assessing object-oriented code quality
D. CK metrics measure class-level properties like coupling, cohesion, inheritance, and method complexity

Answer: D

Q1249. A manager wants to compare productivity across teams using different technologies. What measurement approach is most fair?

A. Compare teams solely on lines of code produced regardless of the programming languages they use
B. Use technology-independent metrics like function points delivered per unit of effort for comparison
C. Use identical metrics thresholds for all teams regardless of their technology stack differences
D. Avoid any cross-team comparison because different technologies make all comparisons meaningless

Answer: B

Q1250. What is the statistical validity concern when using small sample sizes for software metrics?

A. Software metrics do not follow statistical distributions so sample size is completely irrelevant
B. Small sample sizes always produce more accurate statistics than large sample sizes in all cases
C. Sample size has no impact on the statistical validity of software metrics or their conclusions
D. Small samples produce unreliable statistics with wide confidence intervals limiting conclusion validity

Answer: D

Q1251. A large distributed team experiences frequent integration failures. How should their CM practices be improved?

A. Eliminate automated testing to speed up the integration process and reduce pipeline execution time
B. Have each developer maintain a separate repository that is never integrated with other team work
C. Reduce commit frequency to monthly batches so integration issues can be resolved all at once now
D. Implement CI with automated builds and tests, smaller commits, and trunk-based development workflow

Answer: D

Q1252. How does infrastructure as code (IaC) extend traditional configuration management principles?

A. It replaces all software configuration management with manual server setup and configuration work
B. It applies version control, reproducibility, and automation principles to infrastructure provisioning
C. It eliminates the need for version control by automating all infrastructure changes automatically
D. IaC has no relationship with configuration management principles or practices in any context now

Answer: B

Q1253. What are the trade-offs between feature branching and trunk-based development strategies?

A. Trunk-based development cannot work with more than three developers on any given project team
B. Feature branching is always superior to trunk-based development in every project context scenario
C. Feature branches isolate work but delay integration; trunk-based enables faster integration but needs discipline
D. Feature branching and trunk-based development produce identical outcomes with no trade-offs at all

Answer: C

Q1254. A production incident requires an emergency fix while a major release is in progress. How should configuration management handle this?

A. Create a hotfix branch from the production release tag, apply the fix, merge to both main and release branches
B. Apply the fix directly to the production server without any version control tracking or review
C. Cancel the major release entirely and restart the development from scratch to include the fix
D. Delay the emergency fix until the major release is complete to avoid any branching complexity

Answer: A

Q1255. How does GitOps extend configuration management principles to infrastructure and deployment operations?

A. GitOps eliminates the need for any version control in infrastructure management and operations
B. GitOps requires manual deployment processes without any automation or version control involvement
C. GitOps only applies to application code and has no relevance to infrastructure or operations work
D. GitOps uses Git as the single source of truth for infrastructure and deployment configurations now

Answer: D

Q1256. What is the impact of poor configuration management on software project outcomes?

A. It leads to integration failures, lost changes, deployment errors, and inability to reproduce builds
B. Poor CM has no measurable impact on project outcomes because CM is purely administrative overhead
C. Only large projects are affected by poor CM while small projects are completely immune to problems
D. Poor CM only affects the testing phase and has no impact on development or deployment activities

Answer: A

Q1257. How should database schema changes be managed within a configuration management framework?

A. Use versioned migration scripts tracked in version control with automated application during deployment
B. Store schema documentation in email threads rather than in the version control repository system
C. Database schemas should never change after initial deployment regardless of evolving requirements
D. Apply schema changes directly to production databases without any tracking or version control work

Answer: A

Q1258. What is the relationship between configuration management and reproducible builds?

A. CM has no impact on build reproducibility since builds are determined solely by compiler versions
B. Reproducible builds are impossible regardless of how thorough the configuration management practices
C. CM ensures all build inputs are versioned and tracked enabling identical build reproduction every time
D. Reproducible builds only matter for open-source projects and are irrelevant for commercial software

Answer: C

Q1259. How does artifact management complement source code version control in a complete CM strategy?

A. Artifact management replaces source code version control entirely making repositories unnecessary
B. Only source code needs version control; built artifacts never need tracking or management at all
C. Artifact management and source code control serve identical purposes and using both is redundant
D. It tracks versioned build outputs, dependencies, and deployable packages alongside source code control

Answer: D

Q1260. A team needs to maintain multiple product versions simultaneously for different customers. What CM strategy supports this?

A. Long-lived release branches with selective backporting of changes between supported version branches
B. Maintain separate codebases for each customer version without any shared source code between them
C. Use a single branch and deploy different versions by modifying configuration at deployment time only
D. Force all customers to use the same version and reject any requests for version-specific support

Answer: A

Q1261. A project manager discovers that the team's key architect is considering leaving. How should this personnel risk be managed?

A. Ignore the risk because the architect has not officially resigned from the project team yet now
B. Immediately terminate the architect to control the timing of their departure from the team work
C. Prevent the architect from communicating with other team members to reduce knowledge dependency
D. Document critical knowledge, cross-train team members, and develop a succession plan immediately

Answer: D

Q1262. How does Monte Carlo simulation enhance risk assessment compared to deterministic estimation methods?

A. It models uncertainty by running thousands of simulations to produce probability distributions of outcomes
B. Monte Carlo simulation only works for financial risk assessment and not for software project risks
C. It provides a single exact answer that is always more accurate than any probabilistic approach used
D. It eliminates all uncertainty from risk assessment by computing the exact outcome every single time

Answer: A

Q1263. A project depends on a third-party API that may be deprecated. How should this dependency risk be assessed and managed?

A. Build the entire application tightly coupled to the API without any abstraction or fallback plans
B. Ignore the deprecation risk because third-party providers never actually remove their APIs from use
C. Wait until the API is actually deprecated before taking any action to address the dependency risk
D. Create abstraction layers, monitor API provider communications, and plan migration alternatives early

Answer: D

Q1264. What is the relationship between risk appetite and risk tolerance in organizational risk management?

A. Risk appetite and risk tolerance are identical concepts with no difference in their definitions at all
B. Risk appetite refers to specific measurable thresholds while tolerance is a general attitude toward risk
C. Risk appetite is the overall willingness to take risk; tolerance is the acceptable variation around objectives
D. Organizations should have zero risk appetite and zero risk tolerance to ensure project success always

Answer: C

Q1265. How should risk management be adapted for agile projects compared to traditional plan-driven projects?

A. Agile projects have no risks because the iterative nature eliminates all uncertainty automatically now
B. Apply identical heavyweight risk management processes regardless of whether the project is agile
C. Integrate risk identification into sprint planning and retrospectives with continuous lightweight assessment
D. Perform risk management only once at project start and never revisit risks throughout agile sprints

Answer: C

Q1266. A software product must launch on a fixed date for a regulatory deadline. How does this constraint affect risk management?

A. It elevates schedule risks to critical priority and may require scope negotiation as the main lever
B. Schedule constraints only affect the budget and have no impact on scope or quality risk factors
C. The regulatory deadline should be ignored if the team determines it is not technically achievable
D. Fixed deadlines eliminate all risks because the team will naturally work harder to meet the date

Answer: A

Q1267. What is the difference between known risks, unknown risks, and unknowable risks in project management?

A. Known risks are identified; unknown risks are unidentified but identifiable; unknowable risks cannot be anticipated
B. Unknowable risks are the only category worth managing since known and unknown risks resolve themselves
C. Unknown risks do not exist because thorough analysis will always identify every possible risk event
D. All three categories are identical because every risk can be identified with sufficient analysis effort

Answer: A

Q1268. How does the concept of 'risk budgeting' help project managers allocate resources effectively?

A. Risk budgeting wastes project resources on events that may never occur and should be avoided always
B. All project budget should be allocated to development activities with nothing reserved for risk events
C. It reserves specific time and budget for managing identified risks proportional to their exposure values
D. Risk budgeting only applies to financial institutions and has no relevance to software project work

Answer: C

Q1269. What is the impact of cognitive biases on risk assessment accuracy in software projects?

A. Cognitive biases always improve risk assessment by providing diverse perspectives on potential risks
B. Cognitive biases only affect individual contributors and never influence management risk decisions
C. Project managers are immune to cognitive biases due to their training and professional experience
D. Biases like optimism, anchoring, and groupthink lead to systematic underestimation of project risks

Answer: D

Q1270. A distributed development team faces geopolitical risks across multiple countries. How should these be incorporated into the risk management framework?

A. Only consider geopolitical risks if a team member specifically raises a concern during a meeting
B. Assume all regions have identical risk profiles and apply a single uniform risk assessment globally
C. Assess regulatory, political, and infrastructure risks per region and create contingency plans for each
D. Geopolitical risks are outside the scope of software project risk management and should be ignored

Answer: C

Q1271. A web application stores user passwords in plaintext in the database. What is the proper remediation approach?

A. Encrypt passwords with a symmetric key stored alongside the passwords in the same database table
B. Store passwords in a separate plaintext file on the server instead of in the database tables
C. Continue storing passwords in plaintext because hashing adds unnecessary system complexity now
D. Hash passwords using bcrypt or Argon2 with per-user salts and migrate all existing user records

Answer: D

Q1272. How does the STRIDE threat model categorize security threats?

A. Scalability, Traceability, Reusability, Interoperability, Durability, and Extensibility of design
B. Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, Elevation of privilege
C. Speed, Testing, Reliability, Integration, Deployment, and Efficiency of the system components
D. Storage, Transmission, Retrieval, Indexing, Deletion, and Encryption of application data now

Answer: B

Q1273. A company discovers that a critical zero-day vulnerability affects their production system. What is the recommended incident response?

A. Shut down all company systems permanently until the vulnerability is fixed by the software vendor
B. Post details of the vulnerability publicly to warn other companies before applying any patches
C. Isolate affected systems, assess impact, apply temporary mitigations, develop and deploy a permanent fix
D. Ignore the vulnerability because zero-day means no exploit exists yet in the wild for it now

Answer: C

Q1274. How does the principle of 'security by design' differ from 'security by obscurity'?

A. Security by design has been proven ineffective and all modern systems rely on obscurity instead
B. Security by design builds protection into architecture; obscurity relies on hiding implementation details
C. Security by obscurity is more effective because attackers cannot exploit what they cannot find now
D. Security by design and security by obscurity are identical approaches producing equivalent protection

Answer: B

Q1275. What is the impact of the DevSecOps approach on traditional security engineering practices?

A. It delays all security testing until after production deployment to avoid slowing down development
B. It eliminates all security activities because DevOps automation makes applications inherently secure
C. DevSecOps has no meaningful impact on security practices compared to traditional approaches now
D. It integrates security into every stage of the CI/CD pipeline rather than treating it as a gate only

Answer: D

Q1276. How should an organization implement a zero-trust security architecture?

A. Verify every request regardless of source, enforce least privilege, and assume breach at all times
B. Implement a single perimeter firewall and trust everything inside the corporate network boundary
C. Trust all internal network traffic and only verify requests coming from external network sources
D. Remove all authentication and authorization controls to achieve maximum system simplicity now

Answer: A

Q1277. What is the relationship between security requirements and functional requirements in software engineering?

A. Functional requirements always take priority and security requirements can be ignored if needed
B. Security requirements are completely independent of functional requirements with no interaction now
C. Security requirements constrain how functional requirements are implemented to protect system assets
D. Security requirements only apply to the network layer and never affect application functionality

Answer: C

Q1278. How does supply chain security affect modern software development practices?

A. Supply chain security is irrelevant because all third-party libraries are thoroughly vetted already
B. Only first-party code can contain vulnerabilities since third-party libraries are always secure now
C. Third-party dependencies can introduce vulnerabilities requiring continuous monitoring and auditing
D. Supply chain attacks only target hardware and have no relevance to software development practices

Answer: C

Q1279. What is the role of security testing throughout the software development lifecycle?

A. Early security testing is wasteful because most vulnerabilities are introduced during deployment
B. Security testing should only occur during the final phase before deployment to production servers
C. Security testing at every phase catches vulnerabilities earlier when they are cheaper to fix overall
D. Security testing is unnecessary if the development team follows secure coding guidelines properly

Answer: C

Q1280. How should cryptographic key management be implemented in enterprise software systems?

A. Store encryption keys in plaintext files alongside the encrypted data for easy access and retrieval
B. Use hardware security modules, automated key rotation, and strict access controls for all key materials
C. Use a single encryption key for all data across the entire organization without any rotation ever
D. Hardcode encryption keys directly into the application source code for convenience and reliability

Answer: B

Q1281. What does Lehman's Law of Continuing Growth state?

A. Software always decreases in size
B. Software must continuously grow in functionality to remain satisfactory to users
C. Software never changes after release
D. Growth always improves quality

Answer: B

Q1282. In the context of socio-technical systems, what is an organizational system?

A. A system made only of hardware
B. A broader system that includes social and organizational policies and rules governing how people interact with the technical system
C. A database management system
D. An operating system

Answer: B

Q1283. What is the 'no silver bullet' argument by Fred Brooks?

A. That all software should use silver encryption
B. That there is no single technique that will dramatically improve software productivity by an order of magnitude
C. That only one programming language should be used
D. That testing alone can guarantee quality

Answer: B

Q1284. What distinguishes 'essential complexity' from 'accidental complexity' in software?

A. Essential complexity can be eliminated; accidental cannot
B. Essential complexity is inherent to the problem; accidental complexity arises from the chosen solution approach
C. They are the same concept
D. Essential complexity relates only to hardware

Answer: B

Q1285. What is the Standish Group's CHAOS report primarily known for?

A. Evaluating programming languages
B. Analyzing software project success and failure rates
C. Rating software companies
D. Benchmarking hardware performance

Answer: B

Q1286. Which of the following best describes the concept of 'wicked problems' in software engineering?

A. Problems that involve malicious code
B. Problems that are inherently ill-defined and have no definitive solution
C. Simple problems with obvious solutions
D. Problems caused by poor coding practices

Answer: B

Q1287. What is the principle of 'separation of concerns' in software engineering?

A. Keeping developers separate from testers
B. Dividing a program into distinct sections that each address a separate concern or functionality
C. Separating the budget from the schedule
D. Using different computers for different tasks

Answer: B

Q1288. What is Lehman's Law of Declining Quality?

A. Software quality always improves
B. Software will be perceived as declining in quality unless rigorously maintained and adapted to its operational environment
C. Quality is unrelated to maintenance
D. Only hardware quality declines

Answer: B

Q1289. What role does abstraction play in managing software complexity?

A. It makes software run faster
B. It hides unnecessary details so developers can focus on relevant aspects at each level
C. It eliminates all bugs
D. It replaces the need for testing

Answer: B

Q1290. What is the 'Pareto Principle' as applied to software defects?

A. All modules have equal defect rates
B. Roughly 80% of defects are found in approximately 20% of the modules
C. Defects can never be found
D. Testing eliminates 100% of defects

Answer: B

Q1291. What are the four phases of the Rational Unified Process (RUP)?

A. Plan, Code, Test, Deploy
B. Inception, Elaboration, Construction, Transition
C. Analysis, Design, Code, Maintain
D. Gather, Build, Verify, Release

Answer: B

Q1292. What is the cleanroom software engineering process?

A. Development done in a physically clean room
B. A process emphasizing formal specification, incremental development, and statistical testing to produce near-zero-defect software
C. A process that only uses one programmer
D. A process that avoids all documentation

Answer: B

Q1293. What is the concept of 'process tailoring'?

A. Designing new programming languages
B. Adapting a standard process model to fit specific project needs, constraints, and organizational culture
C. Ignoring all processes
D. Writing code without any process

Answer: B

Q1294. In the Spiral model, what happens during the 'planning' quadrant?

A. Code is written
B. Project scope, schedule, and cost estimates are reviewed and the next iteration is planned based on risk analysis results
C. The software is deployed
D. Users are trained

Answer: B

Q1295. What is the difference between verification and validation in process models?

A. They are the same activity
B. Verification checks if the product is built correctly per specification; validation checks if the right product is built for the user
C. Verification is done after deployment; validation is done before coding
D. Neither involves testing

Answer: B

Q1296. What is a stage-gate process in software development?

A. A process with no checkpoints
B. A process where each phase ends with a formal review (gate) that must be passed before proceeding to the next stage
C. A process that only uses one language
D. An automated deployment pipeline

Answer: B

Q1297. What is the key criticism of prescriptive process models in modern software development?

A. They are too flexible
B. They can be too rigid and document-heavy, failing to adapt to rapidly changing requirements and technologies
C. They produce software too quickly
D. They never include testing

Answer: B

Q1298. What is the Capability Maturity Model Integration (CMMI) and how does it relate to process models?

A. A programming language
B. A framework for assessing and improving organizational process maturity across five levels
C. A type of database
D. A testing tool

Answer: B

Q1299. What is the Cynefin framework and how does it relate to Agile?

A. A programming framework
B. A sense-making model that categorizes problems into domains (simple, complicated, complex, chaotic) to guide the appropriate management approach
C. A database management system
D. A version control system

Answer: B

Q1300. What is the difference between Scrum and Scrumban?

A. There is no difference
B. Scrumban combines Scrum's structure with Kanban's flow-based approach, using WIP limits instead of strict Sprint commitments
C. Scrumban eliminates all Scrum ceremonies
D. Scrumban is only for hardware development

Answer: B

Q1301. What is the concept of 'swarming' in Agile?

A. The team working on many tasks simultaneously
B. Multiple team members collaborating on a single high-priority item to complete it quickly
C. A technique for writing documentation
D. A method for deploying software

Answer: B

Q1302. What is Large-Scale Scrum (LeSS)?

A. A version of Scrum for small teams
B. A framework for applying Scrum principles to multiple teams working on a single product with minimal additional roles or artifacts
C. A testing methodology
D. A documentation standard

Answer: B

Q1303. What is the 'iron triangle' trade-off in Agile versus traditional project management?

A. Agile fixes scope and varies time/cost
B. In traditional management scope is fixed while in Agile time and cost are fixed and scope varies
C. They handle trade-offs identically
D. Agile ignores all constraints

Answer: B

Q1304. What is Behavior-Driven Development (BDD) and how does it extend TDD?

A. BDD replaces all testing
B. BDD extends TDD by using natural language specifications that describe system behavior from the user's perspective
C. BDD is unrelated to testing
D. BDD only tests database operations

Answer: B

Q1305. What is the concept of 'Minimum Viable Product' (MVP) in Agile?

A. The cheapest product possible
B. The smallest version of a product that delivers enough value to early adopters and provides validated learning
C. A product with no features
D. The final release version

Answer: B

Q1306. What is the Spotify model in Agile?

A. A music streaming app
B. An organizational model using Squads, Tribes, Chapters, and Guilds to balance team autonomy with cross-functional alignment
C. A testing framework
D. A deployment strategy

Answer: B

Q1307. What is the Agile concept of 'sustainable pace'?

A. Working overtime every day
B. Maintaining a work rhythm that the team can sustain indefinitely without burnout
C. Releasing software once a year
D. Working only 2 hours per day

Answer: B

Q1308. What are Agile release trains (ARTs) in the Scaled Agile Framework (SAFe)?

A. Physical trains that deliver software
B. Long-lived teams of Agile teams that plan, commit, and execute together in Program Increments
C. A type of version control
D. A deployment tool

Answer: B

Q1309. What is the KAOS methodology in requirements engineering?

A. A methodology for causing chaos
B. A goal-oriented requirements engineering approach that derives requirements from high-level organizational goals
C. A testing technique
D. A coding standard

Answer: B

Q1310. What is the difference between safety requirements and security requirements?

A. They are the same
B. Safety requirements prevent harm to people or environment; security requirements protect against malicious threats
C. Safety is about passwords
D. Security is about physical safety

Answer: B

Q1311. What is the problem of 'requirements creep'?

A. Requirements that are too small
B. The uncontrolled expansion of project scope as new requirements are continuously added without proper change management
C. Requirements that are well-managed
D. A beneficial increase in features

Answer: B

Q1312. What is the i* (iStar) framework in requirements engineering?

A. A programming language
B. An agent-oriented modeling framework that captures stakeholder goals, dependencies, and relationships
C. A testing framework
D. A deployment tool

Answer: B

Q1313. How does ethnography contribute to requirements elicitation?

A. It replaces all other elicitation techniques
B. It reveals implicit requirements by observing users in their actual work environment to understand real practices
C. It only studies ancient cultures
D. It is a testing technique

Answer: B

Q1314. What is the Kano model in requirements analysis?

A. A database model
B. A model that classifies features into must-be, one-dimensional, attractive, indifferent, and reverse categories based on customer satisfaction
C. A process model
D. A testing model

Answer: B

Q1315. What is the concept of 'requirements engineering for self-adaptive systems'?

A. Systems that never change
B. Specifying requirements that allow a system to monitor itself and dynamically adjust its behavior to meet goals
C. Requirements written by AI
D. Systems without any requirements

Answer: B

Q1316. What is the role of natural language processing (NLP) in modern requirements engineering?

A. It has no role
B. NLP tools help detect ambiguities, inconsistencies, and missing information in requirements documents
C. NLP replaces all requirements activities
D. NLP is only used for coding

Answer: B

Q1317. What is the Putnam model (SLIM) used for in software estimation?

A. Estimating physical weight of hardware
B. Using a Rayleigh curve to relate project effort, development time, and software size for effort estimation
C. Estimating employee satisfaction
D. Predicting market trends

Answer: B

Q1318. What is the difference between COCOMO I and COCOMO II?

A. There is no difference
B. COCOMO II accounts for modern development practices including reuse, object-oriented development, and iterative processes
C. COCOMO I is newer
D. COCOMO II only estimates hardware costs

Answer: B

Q1319. What is the Wideband Delphi estimation technique?

A. A single expert making all estimates
B. A consensus-based estimation method where experts independently estimate, discuss differences, and re-estimate iteratively
C. A database query optimization method
D. An automated code generation technique

Answer: B

Q1320. What is the concept of 'planning poker' in Agile estimation?

A. A card game for entertainment
B. A consensus-based estimation technique where team members simultaneously reveal their estimates using cards to avoid anchoring bias
C. A method for assigning tasks randomly
D. A technique for prioritizing bug fixes

Answer: B

Q1321. What is the concept of 'cone of uncertainty' in estimation?

A. A physical cone used in meetings
B. The principle that estimation accuracy improves as the project progresses, with early estimates having 4x variability
C. Uncertainty never decreases
D. Estimates are always accurate from the start

Answer: B

Q1322. What is the Theory of Constraints (TOC) as applied to software project management?

A. A theory about hardware limitations
B. A management philosophy that identifies the most critical limiting factor (bottleneck) and systematically improves it
C. A programming paradigm
D. A database optimization theory

Answer: B

Q1323. What is the concept of 'project velocity' in iterative estimation?

A. The speed of code compilation
B. Using historical throughput data from past iterations to forecast how much work can be completed in future iterations
C. The network speed of the team
D. How fast code is deployed

Answer: B

Q1324. What is Goldratt's Critical Chain Project Management (CCPM)?

A. Managing chains of emails
B. A method that accounts for resource constraints and uses project buffers instead of task-level buffers to protect the schedule
C. A version control strategy
D. A code review technique

Answer: B

Q1325. What is the purpose of post-mortem analysis in project management?

A. Analyzing deceased team members
B. Reviewing a completed project to identify lessons learned, successes, and areas for improvement
C. Analyzing legacy code
D. Testing production systems

Answer: B

Q1326. What is the Visitor design pattern?

A. A pattern for tracking website visitors
B. A pattern that separates an algorithm from the object structure it operates on, allowing new operations without modifying the objects
C. A pattern for user authentication
D. A pattern for logging

Answer: B

Q1327. What is the Abstract Factory design pattern?

A. A factory building blueprint
B. A pattern that provides an interface for creating families of related objects without specifying their concrete classes
C. An abstract art design tool
D. A database abstraction layer

Answer: B

Q1328. What is the Law of Demeter and why is it important?

A. A mathematical law
B. A design principle stating that a method should only call methods on its own object, parameters, created objects, or direct components, reducing coupling
C. A law about documentation
D. A law about data encryption

Answer: B

Q1329. What is the difference between the Bridge and Adapter patterns?

A. They are identical
B. The Bridge pattern separates abstraction from implementation by design; the Adapter makes existing incompatible interfaces work together after the fact
C. Bridge is for databases; Adapter is for files
D. Neither involves interfaces

Answer: B

Q1330. What is the concept of 'design debt' in software engineering?

A. The financial cost of design tools
B. The accumulated cost of shortcuts and suboptimal design decisions that will require future refactoring
C. Debt incurred by buying design software
D. The cost of hiring designers

Answer: B

Q1331. What is the Mediator design pattern?

A. A legal mediation tool
B. A pattern that defines an object that encapsulates how a set of objects interact, promoting loose coupling by preventing direct references
C. A middleware framework
D. A database connector

Answer: B

Q1332. What is the concept of 'design for testability'?

A. Making the UI look like a test
B. Structuring software so that individual components can be easily isolated and tested through techniques like dependency injection
C. Writing tests before design
D. Testing the design document

Answer: B

Q1333. What is the Chain of Responsibility design pattern?

A. An organizational hierarchy chart
B. A pattern that passes a request along a chain of handlers, where each handler decides whether to process it or pass it to the next handler
C. A supply chain management tool
D. A blockchain pattern

Answer: B

Q1334. What is the Architecture Tradeoff Analysis Method (ATAM) used for?

A. Analyzing stock trades
B. Evaluating software architectures against quality attributes by identifying sensitivity points, tradeoff points, and risks
C. Analyzing code performance
D. Trading software licenses

Answer: B

Q1335. What is the concept of 'eventual consistency' in distributed systems?

A. Systems that are always consistent
B. A model where, given enough time without new updates, all replicas of data will converge to the same value
C. Systems that are never consistent
D. A database locking mechanism

Answer: B

Q1336. What does the CAP theorem state?

A. All distributed systems are perfect
B. A distributed system can provide at most two of three guarantees: Consistency, Availability, and Partition tolerance simultaneously
C. Distributed systems have no limitations
D. CAP stands for Code, Application, Program

Answer: B

Q1337. What is the Strangler Fig pattern for migrating legacy systems?

A. A pattern for growing plants
B. A migration strategy that gradually replaces parts of a legacy system with new components until the old system is completely replaced
C. A security vulnerability pattern
D. A testing pattern

Answer: B

Q1338. What is the concept of 'architectural fitness functions'?

A. Exercise routines for architects
B. Automated tests that verify whether an architecture meets its intended quality attributes and constraints over time
C. A measure of code readability
D. A design pattern

Answer: B

Q1339. What is the Saga pattern in microservices?

A. A storytelling framework
B. A pattern for managing distributed transactions across multiple services using a sequence of local transactions with compensating actions
C. A logging framework
D. A deployment pattern

Answer: B

Q1340. What is the concept of 'domain-driven design' (DDD) in architecture?

A. Designing domain names for websites
B. An approach that models software structure and behavior around the business domain using bounded contexts, aggregates, and ubiquitous language
C. A database design approach
D. A UI design methodology

Answer: B

Q1341. What is the Bulkhead pattern in resilient architecture?

A. A ship construction pattern
B. A pattern that isolates system components into pools so that failure in one does not cascade and bring down the entire system
C. A database partitioning strategy
D. A UI layout pattern

Answer: B

Q1342. What is a cell-based architecture?

A. An architecture inspired by biological cells
B. An architecture that groups related microservices into independently deployable and scalable cells, each containing a complete stack
C. A spreadsheet-based architecture
D. A hardware design approach

Answer: B

Q1343. What is the difference between summative and formative usability evaluation?

A. They are the same
B. Formative evaluation identifies problems during design to improve the product; summative evaluation measures overall usability of the finished product
C. Summative is done first
D. Formative evaluation cannot find problems

Answer: B

Q1344. What is the GOMS model in HCI?

A. A graphics rendering model
B. A cognitive modeling method that predicts user performance by analyzing Goals, Operators, Methods, and Selection rules
C. A database model
D. A project management model

Answer: B

Q1345. What is the concept of 'cognitive walkthrough' in usability evaluation?

A. Walking while thinking
B. A usability inspection method where evaluators step through task scenarios to identify learnability problems from a new user's perspective
C. A physical exercise program
D. A code review technique

Answer: B

Q1346. What is Hick's Law and its implication for UI design?

A. A law about screen resolution
B. The time to make a decision increases logarithmically with the number of choices, implying designers should minimize options
C. A law about animation speed
D. A law about data transfer

Answer: B

Q1347. What is the ISO 9241-210 standard about?

A. Database design
B. An international standard providing requirements and recommendations for human-centered design of interactive systems
C. Network protocols
D. Software licensing

Answer: B

Q1348. What is the concept of 'design thinking' in UI/UX?

A. Thinking about design while sleeping
B. A human-centered innovation approach using empathy, ideation, prototyping, and testing to solve complex design problems iteratively
C. A method for writing code
D. A database design technique

Answer: B

Q1349. What is the Weber-Fechner Law in the context of UI design?

A. A law about network latency
B. A psychophysical law stating that the perceived difference between stimuli is proportional to their ratio, affecting how users notice UI changes
C. A database normalization rule
D. A coding standard

Answer: B

Q1350. What is the concept of 'emotional design' as described by Don Norman?

A. Designing software that makes users cry
B. A design philosophy addressing three levels of user experience: visceral (appearance), behavioral (usability), and reflective (self-image)
C. Designing emoticons
D. A psychological test

Answer: B

Q1351. What is the concept of 'defensive programming'?

A. Programming while on defense in sports
B. Writing code that anticipates and handles unexpected inputs, states, and errors gracefully to improve robustness
C. Programming in a secure facility
D. Programming with antivirus software

Answer: B

Q1352. What is the difference between optimistic and pessimistic concurrency control?

A. They produce identical results
B. Optimistic assumes conflicts are rare and checks at commit time; pessimistic locks resources preemptively to prevent conflicts
C. Optimistic always uses locks
D. Pessimistic never uses locks

Answer: B

Q1353. What is the concept of 'immutability' and why is it valuable in implementation?

A. Code that cannot be read
B. Objects whose state cannot be modified after creation, reducing bugs from unintended state changes and simplifying concurrent programming
C. Variables that can change freely
D. A database locking mechanism

Answer: B

Q1354. What is the 'fail-fast' principle in implementation?

A. Making software crash as often as possible
B. Detecting and reporting errors immediately at the point of failure rather than allowing them to propagate and cause obscure failures later
C. Deploying quickly without testing
D. Skipping error handling

Answer: B

Q1355. What is aspect-oriented programming (AOP)?

A. Programming from different angles
B. A programming paradigm that separates cross-cutting concerns like logging, security, and transactions from business logic
C. Programming for aspect ratios
D. A type of object-oriented programming

Answer: B

Q1356. What is the concept of 'continuous delivery' pipeline?

A. A pipe for delivering water
B. An automated software release pipeline that ensures code changes are always in a deployable state through build, test, and staging automation
C. A continuous email notification system
D. A hardware delivery system

Answer: B

Q1357. What is the concept of 'feature flags' in implementation?

A. Physical flags on servers
B. Conditional switches in code that enable or disable features at runtime without deploying new code
C. Flags in a debugging tool
D. Country flags in the UI

Answer: B

Q1358. What is the concept of 'contract-based programming'?

A. Legal contracts for developers
B. A programming approach where functions define preconditions, postconditions, and invariants as formal contracts
C. Freelance programming
D. A licensing scheme

Answer: B

Q1359. What is the strangler pattern applied to codebase modernization?

A. A gardening technique
B. Gradually replacing legacy code by wrapping it with new implementations and redirecting calls until the old code can be safely removed
C. Strangling competition
D. A compression algorithm

Answer: B

Q1360. What is the concept of 'trunk-based development'?

A. Developing software on a tree trunk
B. A branching strategy where all developers commit to a single main branch frequently, using short-lived feature branches
C. Using elephant trunks as input devices
D. A storage management technique

Answer: B

Q1361. What is mutation testing and how does it assess test quality?

A. Testing with mutated viruses
B. A technique that introduces small changes (mutants) to the source code and checks whether existing tests detect these changes
C. Genetic testing of software
D. Testing software mutations over time

Answer: B

Q1362. What is the concept of 'combinatorial testing' (pairwise testing)?

A. Testing combinations of locks
B. A technique that tests interactions between parameters by covering all pairwise combinations, significantly reducing the number of test cases needed
C. Combining two test suites
D. Testing with two testers

Answer: B

Q1363. What is symbolic execution in testing?

A. Testing with symbols instead of code
B. A technique that analyzes programs by using symbolic values instead of concrete inputs to explore multiple execution paths simultaneously
C. A type of code signing
D. Writing tests using special symbols

Answer: B

Q1364. What is the concept of 'chaos engineering' in testing?

A. Creating chaos in the testing team
B. Deliberately injecting failures into a production system to test its resilience and identify weaknesses before real failures occur
C. Disorganized testing practices
D. Testing without any plan

Answer: B

Q1365. What is the oracle problem in software testing?

A. A problem with Oracle databases
B. The difficulty of determining the correct expected output for a given test input, especially for complex or novel systems
C. A licensing issue
D. A performance problem

Answer: B

Q1366. What is metamorphic testing?

A. Testing software transformation
B. A technique that addresses the oracle problem by defining relationships between inputs and outputs across multiple test executions
C. Testing software evolution
D. Testing user interface transitions

Answer: B

Q1367. What is the concept of 'shift-left testing'?

A. Moving test results to the left column
B. Integrating testing activities earlier in the software development lifecycle to find defects sooner when they are cheaper to fix
C. Testing only left-aligned text
D. A specific testing tool

Answer: B

Q1368. What is fuzzing (fuzz testing)?

A. Making software fuzzy
B. An automated testing technique that provides invalid, unexpected, or random data as inputs to discover vulnerabilities and crashes
C. Testing with blurred images
D. A type of performance test

Answer: B

Q1369. What is Lehman's Law of Increasing Complexity and how does it affect maintenance?

A. Complexity always decreases
B. As software evolves, its complexity increases unless deliberate effort is made to reduce it, making maintenance progressively harder
C. Complexity has no impact on maintenance
D. Only new software is complex

Answer: B

Q1370. What is the concept of 'software rejuvenation'?

A. Making old software look new visually
B. Proactively restarting or refreshing software systems to prevent failures caused by accumulated degradation such as memory leaks
C. Celebrating a software's anniversary
D. Installing software updates

Answer: B

Q1371. What is the 'ripple effect' in software maintenance?

A. An animation effect in the UI
B. The phenomenon where a change to one part of the system causes unintended changes or failures in other parts
C. A water-themed design pattern
D. A network propagation technique

Answer: B

Q1372. What is the concept of 'technical debt management' in maintenance?

A. Managing financial debt
B. Strategically identifying, measuring, and prioritizing the repayment of accumulated design and code shortcuts to maintain system health
C. Managing team debt
D. Managing hardware debt

Answer: B

Q1373. What is the difference between big-bang and incremental migration for legacy systems?

A. They are the same approach
B. Big-bang replaces the entire legacy system at once; incremental migration gradually replaces parts while both systems run in parallel
C. Big-bang is always safer
D. Incremental migration is never used

Answer: B

Q1374. What is the concept of 'software archeology' in maintenance?

A. Digging up old hardware
B. The practice of investigating and understanding legacy code through code reading, tooling, and historical analysis when documentation is missing
C. Collecting vintage software
D. Building a software museum

Answer: B

Q1375. What is the concept of 'planned obsolescence' versus 'unplanned obsolescence' in software?

A. They are identical
B. Planned obsolescence is an intentional end-of-life strategy; unplanned obsolescence occurs when external changes make the software unusable unexpectedly
C. Neither affects maintenance
D. Obsolescence never happens in software

Answer: B

Q1376. What is the concept of 'evolution-aware software design'?

A. Designing software inspired by biological evolution
B. Designing software from the start with the expectation of future changes, making modification points explicit and changes easier
C. Using evolutionary algorithms
D. Designing without any plan

Answer: B

Q1377. What is the challenge of maintaining systems with undocumented dependencies?

A. There is no challenge
B. Hidden dependencies can cause unexpected cascading failures when changes are made, as maintainers cannot anticipate which components will be affected
C. Undocumented dependencies improve performance
D. Dependencies do not affect maintenance

Answer: B

Q1378. What is the concept of 'continuous maintenance' in DevOps?

A. Maintenance done once a year
B. An approach where maintenance activities are integrated into daily development workflows through automation, monitoring, and continuous improvement
C. Maintaining only during business hours
D. Outsourcing all maintenance

Answer: B

Q1379. What is the GQM (Goal-Question-Metric) paradigm and how is it applied?

A. A gaming metric
B. A structured approach where quality goals are defined, questions identify what needs measurement, and metrics provide the answers
C. A graphics quality metric
D. A database query method

Answer: B

Q1380. What is the concept of 'quality culture' in an organization?

A. A culture of testing everything
B. An organizational environment where quality is valued by everyone, built into processes, and continuously improved as a shared responsibility
C. A culture of writing documentation
D. A culture of using expensive tools

Answer: B

Q1381. What is the Rayleigh model used for in defect prediction?

A. Predicting weather patterns
B. A statistical model that describes the expected distribution of defect discovery rates over time during development and testing
C. Predicting stock prices
D. Predicting network traffic

Answer: B

Q1382. What is the concept of 'defect amplification' in the software process?

A. Making defects louder
B. The phenomenon where undetected defects in one phase generate additional defects in subsequent phases, multiplying the total defect count
C. Amplifying error messages
D. Increasing test coverage

Answer: B

Q1383. What is Cleanroom software engineering in the context of QA?

A. Developing in a sterile room
B. A quality-focused approach using formal specification, structured programming, and statistical usage testing to achieve near-zero-defect software
C. A method of cleaning code
D. Developing with air purifiers

Answer: B

Q1384. What is the concept of 'quality attribute scenarios' in quality assessment?

A. Movie scripts about quality
B. Structured descriptions that specify quality attribute requirements through source, stimulus, environment, artifact, response, and response measure
C. Testing scenarios only
D. User story templates

Answer: B

Q1385. What is the relationship between process maturity and defect rates?

A. There is no relationship
B. Higher process maturity levels are generally associated with lower defect rates and more predictable quality outcomes
C. Higher maturity means more defects
D. Process maturity is irrelevant to defects

Answer: B

Q1386. What is the concept of 'total quality management' (TQM) applied to software?

A. A testing management tool
B. A management philosophy emphasizing continuous improvement, customer focus, and involvement of all employees in quality across the entire organization
C. A quality certification
D. A project management methodology

Answer: B

Q1387. What is the Coupling Between Objects (CBO) metric in the CK suite?

A. The number of physical connections between servers
B. The count of other classes to which a class is coupled through method calls, variable access, or inheritance, indicating its dependency level
C. The number of database connections
D. The number of API calls

Answer: B

Q1388. What is the concept of 'measurement dysfunction' in software metrics?

A. Broken measurement tools
B. The phenomenon where measuring a specific attribute causes people to optimize for that metric at the expense of actual goals
C. Metrics that produce wrong numbers
D. Dysfunction in the measurement team

Answer: B

Q1389. What is Halstead's concept of 'program vocabulary' and 'program length'?

A. The number of English words in comments
B. Vocabulary is the number of distinct operators and operands; length is the total number of operators and operands used in the program
C. The number of variable names
D. The number of code files

Answer: B

Q1390. What is the concept of 'technical debt ratio' as a metric?

A. The company's financial debt ratio
B. The ratio of the cost to fix all code issues to the total cost of developing the code, indicating the relative amount of accumulated technical shortcuts
C. The ratio of developers to managers
D. The ratio of tests to code

Answer: B

Q1391. What is the concept of 'object-oriented design metrics' and how do they differ from traditional metrics?

A. They are the same as traditional metrics
B. OO metrics measure class-specific attributes like inheritance depth, coupling, and cohesion that traditional metrics like LOC and cyclomatic complexity do not capture
C. OO metrics only measure code size
D. Traditional metrics are always superior

Answer: B

Q1392. What is the concept of 'code churn' as a metric?

A. The sound of a hard drive
B. The frequency and volume of code changes over time, where high churn in a module often correlates with higher defect rates
C. The number of developers who leave
D. The compilation time

Answer: B

Q1393. What is the purpose of 'software reliability growth models'?

A. Growing software like plants
B. Mathematical models that use defect discovery data to predict remaining defects and estimate when software will reach a target reliability level
C. Models for increasing server capacity
D. Models for team growth

Answer: B

Q1394. What is the concept of 'metric validation' in software measurement?

A. Checking if numbers add up correctly
B. The process of ensuring that a metric actually measures what it claims to measure and is useful for its intended purpose
C. Validating test results
D. Checking code syntax

Answer: B

Q1395. What is the concept of 'configuration identification' in SCM?

A. Identifying configuration files
B. The process of selecting configuration items, defining their relationships, and establishing a naming and labeling scheme for unique identification
C. Identifying team members
D. Identifying hardware components

Answer: B

Q1396. What is the concept of 'infrastructure as code' (IaC) in configuration management?

A. Building physical infrastructure
B. Managing and provisioning computing infrastructure through machine-readable definition files rather than manual configuration
C. Writing code on paper
D. A type of programming language

Answer: B

Q1397. What is the concept of 'immutable infrastructure' in modern configuration management?

A. Infrastructure that is impossible to change
B. A practice where servers are never modified after deployment; any change requires building and deploying new instances from scratch
C. Infrastructure made of unbreakable materials
D. A type of hardware

Answer: B

Q1398. What is a Software Bill of Materials (SBOM) and why is it important?

A. A bill for software purchases
B. A formal, machine-readable inventory of all components, libraries, and dependencies in a software product for security and compliance tracking
C. A financial document
D. An employee roster

Answer: B

Q1399. What is the concept of 'GitOps' as a configuration management approach?

A. Using Git for social media
B. An operational model where Git repositories serve as the single source of truth for declarative infrastructure and application configuration
C. A Git tutorial series
D. A Git plugin

Answer: B

Q1400. What is the difference between 'feature toggles' and 'feature branches' as configuration strategies?

A. They are identical
B. Feature branches isolate work in separate code branches; feature toggles allow incomplete features to exist in the main branch but be disabled at runtime
C. Feature toggles never work
D. Feature branches are always better

Answer: B

Q1401. What is the concept of 'configuration drift' and how is it managed?

A. Physical movement of servers
B. The gradual divergence of actual system configurations from their intended state, managed through automated monitoring and remediation
C. Network signal drift
D. Time zone changes

Answer: B

Q1402. What is the concept of 'dependency management' in configuration management?

A. Managing employee dependencies
B. Systematically tracking, resolving, and updating third-party libraries and components that a software project depends on
C. Managing database dependencies
D. Managing hardware dependencies

Answer: B

Q1403. What is the concept of 'blue-green deployment' in release management?

A. Using blue and green color themes
B. A deployment strategy using two identical production environments where traffic is switched from the current (blue) to the new (green) version
C. A testing strategy using colors
D. A UI design technique

Answer: B

Q1404. What is canary deployment and how does it reduce release risk?

A. Deploying software to canary birds
B. Gradually rolling out changes to a small subset of users before deploying to the entire infrastructure, allowing early detection of issues
C. A type of load testing
D. A security scanning technique

Answer: B

Q1405. What is the Expected Monetary Value (EMV) technique in risk analysis?

A. The expected salary of the project manager
B. A quantitative technique that calculates the average outcome by multiplying each possible outcome's value by its probability and summing all results
C. The expected project budget
D. The expected revenue from the project

Answer: B

Q1406. What is a decision tree in the context of risk management?

A. A tree used for making physical decisions
B. A graphical tool that models decision options, chance events, probabilities, and outcomes to evaluate the expected value of different risk response strategies
C. A type of data structure
D. A tree in a garden used for thinking

Answer: B

Q1407. What is Boehm's Top 10 Risk Items approach?

A. A list of the top 10 most expensive software products
B. A risk management method that continuously tracks the top 10 project risks, ensuring management attention is focused on the most critical risks
C. A list of the top 10 programming languages
D. A ranking of the top 10 developers

Answer: B

Q1408. What is the concept of 'risk appetite' versus 'risk tolerance'?

A. They are identical concepts
B. Risk appetite is the broad level of risk an organization is willing to accept to achieve goals; risk tolerance is the specific acceptable variation from a particular objective
C. Risk appetite is about food preferences
D. Risk tolerance is about physical pain

Answer: B

Q1409. What is the concept of 'anti-fragile' systems in risk management?

A. Systems that break easily
B. Systems that actually improve and become stronger when exposed to stressors, volatility, and disorder
C. Fragile systems that are not working
D. Systems with no risk

Answer: B

Q1410. What is the concept of 'risk burndown' in Agile projects?

A. Burning risk documents
B. A chart tracking the total risk exposure over time, showing whether the project's overall risk is decreasing as planned
C. A risk that causes burnout
D. Deleting risk entries from the backlog

Answer: B

Q1411. What is the Monte Carlo simulation used for in risk analysis?

A. Simulating casino gambling
B. Running thousands of simulations with randomly varied risk parameters to produce a probability distribution of possible project outcomes
C. Simulating weather patterns
D. Simulating network traffic

Answer: B

Q1412. What is the concept of 'residual risk' in risk management?

A. Risk that has been completely eliminated
B. The remaining level of risk after risk response strategies have been applied, which must be documented and monitored
C. Risk from restaurant software
D. The first risk identified

Answer: B

Q1413. What is the STRIDE threat model and how is each element addressed?

A. A walking exercise program
B. A framework categorizing threats as Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege, each with specific countermeasures
C. A software development methodology
D. A testing framework

Answer: B

Q1414. What is the concept of 'zero trust architecture'?

A. Trusting no one in the office
B. A security model that never implicitly trusts any entity inside or outside the network and requires continuous verification of every access request
C. An architecture with zero features
D. An architecture that trusts everything

Answer: B

Q1415. What is a supply chain attack in software security?

A. Attacking a shipping company
B. An attack that compromises a trusted third-party component, library, or tool in the software supply chain to inject malicious code into dependent systems
C. Attacking a grocery store chain
D. A logistics attack

Answer: B

Q1416. What is the concept of 'security by design' (Secure SDLC)?

A. Designing secure buildings
B. Integrating security practices and considerations into every phase of the software development lifecycle rather than adding them as an afterthought
C. Designing security guard uniforms
D. A type of physical security

Answer: B

Q1417. What is the concept of 'least privilege escalation paths' in security analysis?

A. The shortest path to a promotion
B. Analyzing how an attacker with minimal initial access could chain together vulnerabilities or misconfigurations to escalate their privileges to a higher level
C. Walking the shortest path in an office
D. A career development path

Answer: B

Q1418. What is the concept of 'attack surface' in security engineering?

A. The physical surface of a computer
B. The total set of points in a system where an attacker could potentially enter, extract data, or cause damage
C. A type of computer monitor
D. A mouse pad surface

Answer: B

Q1419. What is homomorphic encryption?

A. Encryption that makes all data look the same
B. A form of encryption that allows computations to be performed on encrypted data without decrypting it first, preserving privacy during processing
C. Simple symmetric encryption
D. A type of hash function

Answer: B

Q1420. What is the concept of 'secure coding guidelines' like CERT and CWE?

A. Guidelines for physical building security
B. Standardized catalogs of coding vulnerabilities (CWE) and best practices (CERT) that help developers write code resistant to common security flaws
C. Guidelines for writing fast code
D. Guidelines for team management

Answer: B

Q1421. What is the concept of 'DevSecOps' and how does it differ from traditional security approaches?

A. A type of firewall
B. An approach that integrates security practices directly into the DevOps pipeline, making security automated, continuous, and a shared responsibility rather than a late-stage gate
C. A developer social network
D. A security certification

Answer: B

Q1422. What is the Unified Software Development Process and how does it differ from the Rational Unified Process (RUP)?

A. The Unified Process is a generic framework while RUP is a specific commercial implementation with detailed guidance and tools
B. RUP is a generic framework while the Unified Process is a commercial product
C. They are identical processes with different names used by different companies
D. The Unified Process supports only agile methods while RUP supports only plan-driven methods

Answer: A

Q1423. How does the concept of 'technical debt' influence the selection and adaptation of software process models over time?

A. Technical debt has no effect on process model selection since it is a coding-level concern
B. Process models should never be changed once selected regardless of technical debt
C. Accumulated technical debt may force teams to adopt more structured processes that include explicit refactoring and quality improvement phases
D. Technical debt only affects waterfall projects and has no impact on iterative processes

Answer: C

Q1424. How does the problem frames approach by Michael Jackson differ from traditional requirements analysis methods?

A. It focuses on identifying recurring problem structures in the real world rather than modeling solutions directly
B. It eliminates the need for stakeholder involvement in requirements
C. It is identical to use case modeling but with different terminology
D. It only works for web-based applications and cannot be applied to embedded systems

Answer: A

Q1425. What role does requirements prioritization using the Wiegers' method play in managing conflicting stakeholder demands?

A. It eliminates all conflicts by automatically removing low-priority requirements
B. It uses relative weighting of benefit, penalty, cost, and risk to create an objective priority score that facilitates negotiation
C. It assigns priority based solely on the seniority of the stakeholder who proposed the requirement
D. It defers all prioritization decisions until after implementation is complete

Answer: B

Q1426. How does Little's Law apply to software project management for predicting delivery timelines?

A. It states that adding more people always reduces delivery time proportionally
B. It establishes that average cycle time equals work in progress divided by throughput, helping teams predict delivery based on current workflow
C. It proves that all software projects will exceed their initial estimates by exactly 50%
D. It only applies to manufacturing and has no relevance to software projects

Answer: B

Q1427. What is the Flyweight design pattern and when is it most beneficial?

A. A pattern that ensures a class has only one instance with global access
B. A pattern that uses sharing to support large numbers of fine-grained objects efficiently by separating intrinsic and extrinsic state
C. A pattern that defines a skeleton algorithm in a base class with steps deferred to subclasses
D. A pattern for creating families of related objects without specifying concrete classes

Answer: B

Q1428. How does the concept of 'connascence' extend the traditional understanding of coupling in software design?

A. Connascence is simply another name for coupling with no additional insight
B. Connascence categorizes coupling into static and dynamic forms with varying strengths, providing a more nuanced framework for evaluating design quality
C. Connascence eliminates the need to consider cohesion in design evaluation
D. Connascence only applies to functional programming and not object-oriented design

Answer: B

Q1429. What is the concept of 'architectural quanta' in evolutionary architecture and how does it affect deployment?

A. It refers to the smallest unit of source code that can be compiled independently
B. It is an independently deployable artifact with high functional cohesion and synchronous connascence, defining the boundary for architectural characteristics
C. It is a measurement unit for counting the number of microservices in an architecture
D. It is a quantum computing concept that has no relevance to software architecture

Answer: B

Q1430. How does the KLM-GOMS (Keystroke-Level Model) predict user task completion time?

A. It measures the actual time users take and averages the results across a sample group
B. It sums estimated times for basic operations including keystrokes, pointing, homing, drawing, mental preparation, and system response
C. It uses machine learning algorithms trained on historical user data
D. It relies solely on subjective user satisfaction surveys

Answer: B

Q1431. How does the concept of 'satisficing' in UI design differ from 'optimizing' and what are its implications?

A. Satisficing means users always find the best option while optimizing means they settle for less
B. Users often choose the first option that meets their minimum criteria rather than evaluating all options for the best one, which means interfaces should make good-enough options easy to find
C. Satisficing and optimizing are identical concepts in UI design
D. Satisficing only applies to expert users while novice users always optimize their choices

Answer: B

Q1432. What is concolic testing and how does it combine concrete and symbolic execution?

A. It is a form of manual testing that combines interviews and observations
B. It simultaneously performs concrete execution with actual inputs and symbolic execution with symbolic values to systematically explore execution paths
C. It tests only the user interface using both automated and manual approaches
D. It is an obsolete testing technique replaced entirely by fuzz testing

Answer: B

Q1433. What is the 'all-pairs' or pairwise testing technique and why is it effective for combinatorial testing?

A. It requires testing every possible combination of all input parameters exhaustively
B. It ensures that every pair of input parameter values is tested together at least once, significantly reducing test cases while catching most interaction faults
C. It tests only two input parameters and ignores all others
D. It is a technique where two testers review the same test case for accuracy

Answer: B

Q1434. What is the Orthogonal Defect Classification (ODC) system and how does it improve quality analysis?

A. It is a tool for automatically fixing defects in source code
B. It classifies defects along independent dimensions (type, trigger, impact) to provide actionable feedback about the development process
C. It is a testing technique that replaces all other forms of quality assurance
D. It measures only the number of defects without any classification

Answer: B

Q1435. How does the concept of 'quality attributes trade-off' affect architectural decisions in quality assurance?

A. Quality attributes never conflict with each other, so trade-offs are unnecessary
B. Improving one quality attribute (e.g., security) can negatively impact another (e.g., performance), requiring deliberate architectural trade-off analysis
C. Trade-offs only apply during testing and have no effect on architecture
D. All quality attributes can be maximized simultaneously with sufficient budget

Answer: B

Q1436. What is the Response for a Class (RFC) metric in the CK metrics suite and what are its implications?

A. It counts the number of HTTP responses a web class can generate
B. It counts the set of methods that can potentially be executed in response to a message received by an object of the class, indicating testing complexity
C. It measures the average response time of a class in milliseconds
D. It counts the number of comment lines in a class file

Answer: B

Q1437. How does the concept of 'metric entropy' relate to software measurement programs?

A. Metric entropy means all metrics become more accurate over time as data accumulates
B. As measurement programs mature, collected metrics can become stale, gamed, or lose their diagnostic power, requiring periodic reassessment of what is measured and why
C. Metric entropy is a thermodynamic law that has no application to software engineering
D. It ensures that all metrics remain stable and never need to be updated or replaced

Answer: B

Q1438. How does the concept of 'black swan events' challenge traditional risk management approaches in software projects?

A. Black swan events are easily predicted by standard risk identification techniques
B. These are highly improbable, high-impact events that lie outside normal expectations and cannot be predicted by historical data, requiring resilience-based rather than prediction-based risk strategies
C. Black swan events only affect financial industries and have no relevance to software
D. Traditional risk management already fully accounts for black swan events through contingency planning

Answer: B

Q1439. What is the difference between inherent risk and residual risk, and why is this distinction critical for risk management?

A. Inherent and residual risk are synonymous terms used interchangeably in all contexts
B. Inherent risk is the risk level before any controls or mitigations are applied, while residual risk is the remaining risk after mitigation measures are implemented
C. Residual risk is always zero after proper mitigation strategies are applied
D. Inherent risk only applies to new projects while residual risk applies only to maintenance projects

Answer: B

Q1440. What is the DREAD risk assessment model and how does it quantify security threats?

A. DREAD is a psychological assessment of how much developers fear certain vulnerabilities
B. DREAD rates threats on five dimensions: Damage potential, Reproducibility, Exploitability, Affected users, and Discoverability, each scored to calculate an overall threat rating
C. DREAD is a network protocol for secure data transmission
D. DREAD is a testing framework that automatically detects all security vulnerabilities

Answer: B