Q1: What is Separation of Concerns (SoC), and why is it important in system design?

A. It ensures each module focuses on a specific responsibility.

B. It enables system design to scale without constraints.

C. It prevents redundancy in database management.

D. It eliminates the need for modularity in design.

Answer: A

Q2: Define Modularity. How does it benefit software architecture?

A. Modularity simplifies testing and improves maintainability.

B. Modularity increases system complexity.

C. Modularity eliminates dependencies entirely.

D. Modularity ensures tight coupling.

Answer: A

Q3: Explain the difference between vertical and horizontal scaling.

A. Vertical scaling adds resources to a single node, while horizontal scaling adds more nodes.

B. Horizontal scaling always uses a database, while vertical scaling does not.

C. Vertical scaling adds more nodes, while horizontal scaling replaces nodes.

D. Horizontal scaling is the same as modularity.

Answer: A

Q4: What is encapsulation, and how does it support maintainability?

A. Encapsulation hides the internal implementation of a module to promote loose coupling.

B. Encapsulation removes the need for APIs.

C. Encapsulation increases dependency between components.

D. Encapsulation eliminates the need for modular systems.

Answer: A

Q5: List the key components of the MVC design pattern.

A. Model, View, Controller

B. Module, Visuals, Compiler

C. Methods, Variables, Components

D. Management, Validation, Coordination

Answer: A

Q6: What is the purpose of a Singleton design pattern?

A. The Singleton design pattern ensures only one instance of a class exists at a time.

B. It allows multiple instances of a class.

C. It separates concerns between multiple modules.

D. It promotes high cohesion between classes.

Answer: A

Q7: Describe the concept of an API Gateway in microservices.

A. It handles data storage.

B. It routes requests and manages authentication.

C. It eliminates service discovery.

D. It optimizes memory usage.

Answer: B

Q8: What are the main types of scaling in system design?

A. Vertical and horizontal scaling are the two main types of scaling in system design.

B. Database scaling and process scaling are the two types.

C. Functional scaling is the only scaling type.

D. Modular scaling is the only scaling type.

Answer: A

Q9: Define the Zachman Framework's rows and columns.

A. Rows represent perspectives, and columns represent interrogatives.

B. Rows represent taxonomies, and columns represent databases.

C. Rows represent entities, and columns represent constraints.

D. Rows represent use cases, and columns represent interactions.

Answer: A

Q10: What are the phases of TOGAF's Architecture Development Method (ADM)?

A. Preliminary, Vision, and Business Architecture are part of the ADM phases.

B. Only Preliminary and Vision are included.

C. ADM phases exclude Business Architecture.

D. The ADM phases are unstructured.

Answer: A

Q11: How does Loose Coupling and High Cohesion improve system maintainability?

A. Loose coupling minimizes dependencies between modules, while high cohesion ensures module focus.

B. Loose coupling ensures tight integration.

C. High cohesion increases dependencies.

D. Both eliminate the need for modular systems.

Answer: A

Q12: What strategies can you use to design a modular system in a large enterprise application?

A. Use interfaces and design patterns.

B. Combine unrelated responsibilities.

C. Avoid separation of concerns.

D. Rely entirely on vertical scaling.

Answer: A

Q13: Explain the difference between Adapter and Decorator design patterns with examples.

A. Adapter translates interfaces, while Decorator adds functionality.

B. Adapter adds new behavior, while Decorator optimizes memory.

C. Both are identical in their purpose.

D. Adapter handles database queries, while Decorator manages APIs.

Answer: A

Q14: What is a Service Mesh, and when would you use it in microservices architecture?

A. A Service Mesh provides secure communication and traffic management between services.

B. A Service Mesh eliminates the need for APIs.

C. A Service Mesh directly replaces microservices.

D. A Service Mesh manages only database interactions.

Answer: A

Q15: How would you use Gap Analysis in TOGAF to identify architectural improvements?

A. Compare the baseline and target architectures.

B. Focus exclusively on stakeholder interviews.

C. Identify legacy systems only.

D. Ignore business processes.

Answer: A

Q16: Compare TOGAF's ADM and Zachman Framework. When would you use each?

A. TOGAF's ADM provides a process-driven approach, while Zachman offers a taxonomy for architecture.

B. TOGAF uses interrogatives, while Zachman provides processes.

C. Both frameworks are identical.

D. TOGAF is strictly for database design.

Answer: A

Q17: What are the benefits of using Event Sourcing in a distributed system?

A. It stores only the latest state of the application.

B. It records changes as a series of events for better traceability.

C. It prevents the need for centralized logging.

D. It eliminates the use of APIs.

Answer: B

Q18: Explain how Circuit Breakers prevent cascading failures in distributed systems.

A. Circuit Breakers monitor service health and temporarily stop requests to failing components.

B. Circuit Breakers eliminate the need for fault tolerance.

C. Circuit Breakers ensure permanent disconnection.

D. Circuit Breakers optimize memory usage.

Answer: A

Q19: What are the advantages of CQRS (Command Query Responsibility Segregation)?

A. Unified data models for commands and queries.

B. Separation of read and write responsibilities for scalability.

C. Faster execution of commands.

D. Elimination of database constraints.

Answer: B

Q20: How would you implement centralized logging and monitoring in a microservices architecture?

A. Use a logging library in every microservice.

B. Deploy a logging and monitoring solution like ELK stack.

C. Rely on the API Gateway exclusively.

D. Use manual logs and avoid centralized solutions.

Answer: B