

1) A métrica de abstração (A) é usada para avaliar:

- a) A tendência de um componente a mudar.
- b) A proporção entre classes abstratas e as classes públicas de um pacote.
- c) A quantidade de classes concretas em um pacote em relação às classes abstratas.
- d) O grau de reutilização de código em métodos privados.

2) Qual das seguintes afirmações descreve corretamente a relação ideal entre acoplamento e coesão no projeto de software orientado a objetos?

- a) Alto acoplamento e alta coesão são desejáveis para maximizar a reutilização de código.
- b) Alto acoplamento e baixa coesão são ideais para garantir a flexibilidade e a manutenibilidade do sistema.
- c) Baixo acoplamento e baixa coesão são preferíveis, pois permitem que as classes sejam independentes e facilmente substituíveis.
- d) Baixo acoplamento e alta coesão são geralmente considerados os princípios de design mais desejáveis, pois promovem sistemas mais robustos e fáceis de manter.

3) O que é consciência em termos de design de software?

- a) Uma classe/módulo que tem sua funcionalidade distribuída por várias partes de um sistema, dificultando o seu entendimento.
- b) O relacionamento entre os elementos de código, de tal forma que, a modificação de um requer a atualização do outro.
- c) Quando uma classe ou módulo depende fortemente de outros para a execução de suas funcionalidades.
- d) O comportamento de um sistema altamente dependente de uma sequência específica de execuções de métodos.

4) Considere um sistema composto por dois módulos A e B. Sabemos que há uma dependência cíclica entre eles. O que isso significa?

- a) O módulo A importa o módulo B, mas o inverso não ocorre.
- b) O módulo A depende do módulo B, que por sua vez depende do módulo A.
- c) Os módulos A e B são totalmente independentes.
- d) Ambos os módulos implementam funcionalidades semelhantes e redundantes.

5) Considere o seguinte código JavaScript que implementa um sistema de envio de notificações:

```
class EmailService {
  enviarEmail(destinatario, mensagem) {
    console.log(`Enviando email para ${destinatario}: ${mensagem}`);
    // Lógica real de envio de email aqui
  }
}

class Notificador {
  constructor() {
    // Dependência direta da implementação concreta
    this.servicoDeEmail = new EmailService();
  }

  enviarNotificacao(usuario, mensagem) {
    this.servicoDeEmail.enviarEmail(usuario.email, mensagem);
  }
}

const usuario = { email: 'usuario@example.com' };
const notificador = new Notificador();
notificador.enviarNotificacao(usuario, 'Seu pedido foi processado.');
```

Qual das seguintes modificações no código demonstra a aplicação do Princípio da Inversão de Dependência?

- a) Criar uma interface `ServicoDeNotificacao` com um método `enviar`, e fazer tanto `EmailService` quanto uma nova classe `SMSService` implementarem essa interface. A classe `Notificador` passaria a depender dessa interface em seu construtor.
- b) Alterar a função `enviarEmail` em `EmailService` para aceitar um objeto de usuário em vez de apenas o email.
- c) Remover a classe `EmailService` e incluir a lógica de envio de email diretamente dentro da classe `Notificador`.
- d) Adicionar uma nova classe `SMSService` e fazer a classe `Notificador` instanciar tanto `EmailService` quanto `SMSService`.

6) Considere as seguintes afirmações sobre a complexidade ciclomática de um módulo de software:

- I) Mede o número de caminhos lineares independentes no fluxo de controle de um programa.
- II) Pode ser usada para estimar o número mínimo de casos de teste necessários para cobrir todos os caminhos lógicos.
- III) É calculada com base na contagem de linhas de código.
- IV) Quanto maior for o valor, mais simples e legível é o código.

Quais alternativas estão corretas?

- a) Todas as afirmativas estão corretas.
- b) Apenas as afirmativas III e IV estão corretas.
- c) Apenas as afirmativas I e II estão corretas.
- d) Apenas as afirmativas I e III estão corretas.

7) A métrica de complexidade ciclomática é usada para avaliar o controle de fluxo de um módulo de software. Considerando a interpretação dos seus valores, qual das alternativas abaixo representa uma afirmação correta?

- a) A complexidade ciclomática ideal de um módulo deve ser sempre maior que 15.
- b) Quanto menor for a complexidade ciclomática, menor a necessidade de testes e mais fácil a manutenção.
- c) Um valor de complexidade ciclomática superior a 10 indica alta coesão e bom encapsulamento.
- d) A complexidade ciclomática entre 10 e 20 indica um código relativamente fácil de testar.

8) Sobre os conceitos de agregação e associação em orientação a objetos, analise as afirmações abaixo e assinale a alternativa correta:

- a) Agregação indica uma relação mais fraca que associação, sem qualquer ligação entre os objetos.
- b) Associação é usada apenas para modelar relações "parte-todo", enquanto agregação é usada para relações "um-para-um".
- c) Associação implica sempre que um objeto é destruído junto com o outro.
- d) Agregação é uma forma especial de associação, onde um objeto faz parte de outro, mas pode existir independentemente dele.

9) Sobre os conceitos de associação e dependência entre classes, assinale a alternativa correta:

- a) Associação e dependência são sinônimos na modelagem orientada a objetos.
- b) Associação é uma relação temporária entre objetos, enquanto dependência é permanente.
- c) Associação implica que um objeto mantém uma referência a outro, enquanto dependência significa apenas o uso temporário de outro objeto.
- d) Em uma associação, um objeto usa outro apenas em momentos específicos, sem manter uma referência direta.

10) Qual padrão é usado para garantir que uma classe tenha apenas uma instância?

- a) Factory Method
- b) Singleton
- c) Prototype
- d) Builder

11) Em qual situação o padrão Builder é mais apropriado?

- a) Quando precisamos criar objetos complexos com muitas etapas ou combinações.
- b) Quando queremos notificar vários objetos ao mesmo tempo.
- c) Quando precisamos compartilhar estados comuns entre muitos objetos leves.
- d) Quando queremos clonar objetos.

12) Você tem duas implementações diferentes de um mesmo algoritmo (por exemplo, cálculo de frete: Sedex, PAC). Em tempo de execução, quer escolher qual lógica aplicar.

- a) Decorator
- b) Abstract Factory
- c) Command
- d) Strategy

13) Qual das alternativas melhor descreve o que são requisitos arquiteturais?

- a) Requisitos funcionais que definem as funcionalidades centrais do sistema.
- b) Características de qualidade e restrições que influenciam diretamente as decisões de arquitetura.
- c) Detalhes de implementação específicos de módulos.
- d) Regras de negócio aplicadas ao domínio.

14) Qual cenário representa um requisito arquitetural de escalabilidade?

- a) O sistema deve suportar um aumento de 10x no número de usuários sem perda significativa de desempenho.
- b) O sistema deve armazenar dados com criptografia AES-256.
- c) O sistema deve permitir *rollback* de transações.
- d) O sistema deve ter interface traduzida para 3 idiomas.

15) Um sistema que deve rodar em Windows, Linux e macOS prioriza qual requisito arquitetural?

- a) Disponibilidade
- b) Escalabilidade
- c) Portabilidade
- d) Observabilidade

16) Uma empresa deseja adicionar rapidamente novas funcionalidades ao sistema, sem causar muitos efeitos colaterais. Qual requisito arquitetural deve ser mais valorizado?

- a) Segurança
- b) Escalabilidade horizontal
- c) Confiabilidade
- d) Extensibilidade

17) Em qual estilo arquitetural o sistema é dividido em unidades independentes que se comunicam por mensagens síncronas ou assíncronas?

- a) Cliente-Servidor
- b) Camadas
- c) Microserviços
- d) Microkernel

18) Qual característica é mais associada à arquitetura de microserviços?

- a) Todos os módulos são implantados como uma única unidade executável.
- b) Cada serviço é independente, com seu próprio banco de dados e ciclo de vida.
- c) A comunicação ocorre exclusivamente por chamadas síncronas.
- d) É ideal para aplicações pequenas com requisitos estáveis.

19) Em um sistema distribuído, qual é a principal vantagem de utilizar um serviço de mensageria (como RabbitMQ, Apache Kafka ou Amazon SQS) entre componentes?

- a) Garantir que todos os componentes utilizem a mesma linguagem de programação.
- b) Permitir comunicação síncrona em tempo real com baixa latência.
- c) Eliminar completamente a necessidade de armazenamento persistente.
- d) Prover desacoplamento entre produtores e consumidores de dados, além de suportar comunicação assíncrona.

20) Qual princípio fundamental da arquitetura de microserviços está diretamente relacionado ao fato de que cada microserviço deve possuir seu próprio banco de dados?

- a) Escalabilidade horizontal
- b) Isolamento de domínio e dados
- c) Reuso de código
- d) Centralização de logs

21) Em um sistema de microserviços que adota consistência eventual, qual afirmação é verdadeira?

- a) A consistência eventual elimina a necessidade de testes de integração
- b) Todos os dados são sempre consistentes imediatamente após uma escrita
- c) A consistência é garantida apenas durante janelas de manutenção
- d) Os dados podem ficar temporariamente inconsistentes, mas convergem para um estado consistente ao longo do tempo