Universidade Federal de Pernambuco :: Centro de Informática
Sistemas de Informação :: Engenharia de Software
Prof. Vinicius Cardoso Garcia

INSTRUÇÕES:

Leia as instruções com atenção e cuidado e a seguir, responda o caderno de questões.

- Esta avaliação tem 20 questões objetivas, com 5 alternativas, para um total de 10 pontos com **sua resolução sendo individual e sem consulta**.
- Organize o tempo, a prova tem duração de até 1 hora e 40 minutos.
- Não é permitido **abrir qualquer aba** diferente do formulário do Exercício Escolar de Segunda Chamada durante a realização da avaliação.
- Não é permitido nada **em cima da mesa ou no colo**. Guardem os celulares e demais dispositivos inteligentes, digitais, analógicos e mecânicos no bolso ou na mochila (ou equivalente) e a mesma deve estar no chão.
- Responda as questões no **formulário de resposta**.
- Dúvidas podem ser expostas, **publicamente**, durante os **primeiros 30 minutos** da realização do Exercício Escolar de Segunda Chamada.
- Não é permitido ir ao sanitário durante a realização do exercício, **vá antes.**
- **Entender o enunciado** faz parte da avaliação.

Questão 01 - Qual das seguintes práticas é essencial para garantir a segurança em serviços web?

A) Utilização de padrões de design responsivo em interfaces web.
B) Implementação de protocolos de comunicação síncrona entre serviços.
C) Uso de HTTPS em vez de HTTP para a comunicação segura de dados.
D) Aplicação de técnicas de machine learning para a análise de dados do usuário.
E) Armazenamento de dados em cache para melhorar o desempenho do serviço.

Questão 02 - Qual método HTTP deve ser utilizado em uma API RESTful para atualizar parcialmente um recurso existente?

A) GET
B) POST
C) PUT
D) PATCH
E) DELETE

Questão 03 - Qual é o principal objetivo dos Web Services no contexto de desenvolvimento de software?

A) Facilitar o desenvolvimento de interfaces gráficas para os usuários.
B) Permitir a comunicação entre sistemas heterogêneos utilizando protocolos padronizados.
C) Substituir completamente o uso de APIs tradicionais para integração de sistemas.
D) Minimizar a necessidade de acesso a banco de dados em sistemas distribuídos.
E) Aumentar a segurança de aplicações web por meio de autenticação de usuário.

Questão 04 - Qual é uma vantagem da integração de sistemas em Engenharia de Software?

A) Uma vantagem é facilitar a comunicação entre equipes de desenvolvimento, garantindo que todos os membros da equipe tenham acesso às informações necessárias para realizar seu trabalho.
B) Uma vantagem é garantir que todos os sistemas e aplicações estejam protegidos contra ameaças de segurança, como ataques de hackers e vazamento de dados.
C) Uma vantagem é melhorar a experiência do usuário final ao fornecer interfaces de usuário consistentes em diferentes sistemas e aplicações.

Boa sorte!

D) Uma vantagem é conectar diferentes sistemas e aplicações para que possam compartilhar dados e funcionalidades, permitindo a criação de soluções mais completas e eficientes.

E) Uma vantagem é testar sistemas individualmente antes de integrá-los em um ambiente de produção, garantindo a qualidade e confiabilidade das soluções desenvolvidas.

Questão 05 - Considerando as dificuldades inerentes à estimativa de projetos de software, qual técnica pode ser mais eficaz para lidar com a incerteza e variabilidade ao estimar a duração e os custos de um projeto complexo de software?

A) Utilizar uma abordagem fixa, baseando-se exclusivamente em experiências anteriores semelhantes.

B) Aplicar a técnica do PERT (Program Evaluation and Review Technique), que usa estimativas otimistas, pessimistas e mais prováveis para definir prazos e custos.

C) Adotar a estratégia de Waterfall, detalhando todas as etapas do projeto antecipadamente e seguindo um cronograma rígido.

D) Basear todas as estimativas em padrões de indústria, independentemente das especificidades do projeto.

E) Ignorar a estimativa de custos e focar apenas no tempo, pois é o único aspecto controlável no projeto.

Questão 06 - No contexto de aplicar aprendizado de máquina em engenharia de software, qual das seguintes tarefas seria uma aplicação adequada desta tecnologia?

A) Automatizar a geração de código para sistemas complexos de inteligência artificial.

B) Utilizar algoritmos de aprendizado de máquina para prever e identificar defeitos em software com base em dados históricos.

C) Implementar sistemas de aprendizado de máquina para substituir completamente a necessidade de teste de software.

D) Usar aprendizado de máquina para otimizar automaticamente o layout de interfaces de usuário sem intervenção humana.

E) Aplicar técnicas de aprendizado de máquina para aumentar a velocidade do processador durante a execução de software.

Questão 07 - Considere as seguintes características: escalabilidade horizontal, flexibilidade na estrutura de dados e capacidade de armazenar grandes volumes de dados não estruturados ou semi-estruturados. Qual tipo de banco de dados essas características descrevem melhor?

A) Banco de dados relacional

B) Banco de dados orientado a objetos

C) Banco de dados não relacional

D) Sistema de gerenciamento de banco de dados (SGBD) clássico

E) Banco de dados distribuído

Questão 08 - Quais são duas técnicas comuns para garantir a segurança em serviços web?

A) Criptografia simétrica e autenticação baseada em token.

B) Ataques de força bruta e assinatura digital.

C) Injeção de SQL e autenticação de dois fatores.

D) Controle de acesso e ofuscação de código-fonte.

E) Firewall de aplicação web e prevenção de cross-site scripting (XSS).

Questão 09 - Explique brevemente o que é ESB (Enterprise Service Bus) e como ele facilita a integração de sistemas em uma arquitetura orientada a serviços.

A) ESB é um protocolo de comunicação utilizado para trocar mensagens entre sistemas heterogêneos em uma rede corporativa. Ele facilita a integração de sistemas ao fornecer uma infraestrutura centralizada para gerenciar e rotear as mensagens entre os diferentes sistemas.

B) ESB é uma plataforma de middleware utilizada para conectar sistemas e aplicações distribuídas em uma arquitetura orientada a serviços. Ele facilita a integração de sistemas ao fornecer recursos como transformação de mensagens, roteamento, segurança e monitoramento.

Boa sorte!

C) ESB é uma técnica de criptografia utilizada para proteger a comunicação entre sistemas em uma arquitetura orientada a serviços. Ele facilita a integração de sistemas ao garantir a confidencialidade e integridade dos dados transmitidos.
D) ESB é um padrão de projeto de software utilizado para criar interfaces de usuário consistentes em diferentes sistemas e aplicações. Ele facilita a integração de sistemas ao fornecer uma abordagem padronizada para o desenvolvimento de interfaces de usuário.
E) ESB é uma prática de controle de acesso utilizada para determinar quais usuários têm permissão para acessar determinados recursos em uma arquitetura orientada a serviços. Ele facilita a integração de sistemas ao garantir a segurança da comunicação entre cliente e servidor.

Questão 10 - Em um processo de Integração Contínua (CI), qual é o principal benefício de se integrar regularmente o código ao repositório principal e executar testes automáticos a cada integração?

A) Aumentar a complexidade do código para desafiar a equipe de desenvolvimento.
B) Permitir a entrega contínua de novos recursos ao cliente sem testes.
C) Reduzir o risco de conflitos de merge e identificar rapidamente erros e problemas de compatibilidade.
D) Eliminar completamente a necessidade de revisão de código pelos desenvolvedores.
E) Focar exclusivamente no aumento da velocidade de entrega, independentemente da qualidade do código.

Questão 11 - No contexto de Entrega Contínua (CD), qual é o principal objetivo de automatizar o pipeline de entrega, desde a integração do código até a sua implementação em produção?

A) Focar na automação de testes de segurança para evitar vulnerabilidades no software.
B) Garantir que cada alteração no código seja manualmente revisada por um membro da equipe antes da implementação.
C) Aumentar a frequência de lançamentos, reduzindo o tempo de ciclo e permitindo um feedback mais rápido dos usuários.
D) Excluir a necessidade de testes de qualidade, confiando na automação para detectar todos os erros.
E) Concentrar-se unicamente no desenvolvimento de novas funcionalidades, ignorando a manutenção do software existente.

Questão 12 - No contexto de DevOps, qual das seguintes afirmações melhor descreve a relação entre DevOps e a cultura organizacional em uma empresa de engenharia de software?

A) DevOps é estritamente um conjunto de ferramentas técnicas e não influencia a cultura organizacional.
B) A implementação de DevOps não requer mudanças na cultura organizacional, pois foca apenas em automação e ferramentas.
C) DevOps exige uma transformação cultural que enfatiza a colaboração, a comunicação e a integração entre as equipes de desenvolvimento e operações.
D) DevOps é incompatível com culturas organizacionais que valorizam processos estabelecidos e estruturas hierárquicas.
E) A adoção de DevOps leva automaticamente a uma melhoria na cultura organizacional sem esforços adicionais.

Questão 13 - Qual é a principal diferença entre testes de integração e testes de sistema no ciclo de testes de software, e qual é a importância de cada um desses tipos de teste?

A) Testes de integração são realizados para verificar a segurança do software, enquanto os testes de sistema são para verificar a usabilidade.
B) Testes de integração focam em verificar a interação entre diferentes módulos ou componentes do software, enquanto os testes de sistema avaliam o software como um todo, verificando se ele atende aos requisitos especificados.
C) Testes de integração são sempre automatizados, enquanto os testes de sistema são realizados manualmente.

Boa sorte!

D) Testes de sistema são realizados antes dos testes de integração para identificar problemas gerais antes de focar em componentes específicos.
E) Testes de integração são usados apenas em metodologias ágeis de desenvolvimento, enquanto testes de sistema são usados em abordagens tradicionais.

Questão 14 - Como um engenheiro de software pode efetivamente combinar técnicas de teste estático e dinâmico em um projeto complexo para maximizar a cobertura de teste e a descoberta de defeitos?

A) Utilizar somente técnicas de teste dinâmico, pois são mais eficientes para encontrar erros em tempo de execução.
B) Aplicar técnicas de teste estático inicialmente para aprimorar a qualidade do código e, posteriormente, técnicas de teste dinâmico para validar a funcionalidade e desempenho do software em condições reais.
C) Confiar exclusivamente em teste estático, já que ele pode detectar a maioria dos defeitos sem a necessidade de executar o software.
D) Alternar aleatoriamente entre técnicas de teste estático e dinâmico ao longo do projeto, dependendo da disponibilidade da equipe.
E) Delegar testes estáticos para desenvolvedores e testes dinâmicos para uma equipe separada de controle de qualidade.

Questão 15 - Considerando um projeto de software complexo que envolve múltiplas equipes trabalhando em diferentes componentes, como a garantia e o controle de qualidade podem ser efetivamente gerenciados para assegurar a consistência e a alta qualidade em todas as partes do software?

A) Centralizar todas as atividades de garantia e controle de qualidade em uma única equipe especializada, que revisa todo o trabalho no final de cada ciclo de desenvolvimento.
B) Implementar padrões de qualidade universais, realizar revisões de código cruzadas entre as equipes e utilizar testes de integração automatizados para garantir a compatibilidade e a funcionalidade entre componentes.
C) Limitar as atividades de garantia de qualidade às fases de planejamento e design, confiando que os desenvolvedores mantenham a qualidade durante a codificação.
D) Delegar responsabilidades de controle de qualidade exclusivamente para os líderes de equipe, sem processos de revisão centralizados ou testes cruzados.
E) Utilizar uma abordagem baseada exclusivamente em testes manuais realizados por uma equipe de teste independente após a conclusão do desenvolvimento.

Questão 16 - Qual é a importância do princípio "Fail Fast" em sistemas distribuídos e como ele contribui para a resiliência e a manutenção eficiente de tais sistemas?

A) "Fail Fast" significa que o sistema deve evitar falhas a todo custo, focando em redundâncias massivas para garantir a estabilidade.
B) Este princípio prioriza a execução rápida de todas as operações, mesmo que isso leve a um aumento na taxa de falhas.
C) "Fail Fast" incentiva o sistema a detectar e reportar erros o mais rápido possível, melhorando a resiliência através da rápida identificação e correção de problemas.
D) A abordagem "Fail Fast" sugere que os sistemas devem gradualmente diminuir a funcionalidade em resposta a falhas para prolongar a operacionalidade.
E) O princípio é aplicado apenas no estágio de desenvolvimento e teste, focando em falhas rápidas para acelerar o ciclo de desenvolvimento.

Questão 17 - Qual é a melhor estratégia para gerenciar a refatoração de um sistema legado de grande escala, que tem se tornado cada vez mais difícil de manter e evoluir?

A) Realizar uma refatoração completa e imediata de todo o sistema para modernizá-lo, independentemente dos custos e dos riscos envolvidos.
B) Evitar a refatoração e focar no desenvolvimento de novas funcionalidades, integrando-as ao sistema legado conforme necessário.
C) Implementar uma abordagem de refatoração gradual e sistemática, priorizando componentes críticos e integrando melhorias contínuas ao longo do tempo.

Boa sorte!

D) Substituir completamente o sistema legado por uma nova solução desenvolvida do zero, independentemente da compatibilidade com os processos existentes.
E) Terceirizar a refatoração e a manutenção do sistema legado para concentrar recursos internos no desenvolvimento de novos projetos.

Questão 18 - Na gestão de projetos de software, como a métrica de "taxa de defeitos" pode ser utilizada para melhorar a qualidade do produto final?

A) A taxa de defeitos é usada para avaliar a produtividade dos desenvolvedores, incentivando-os a escrever mais código em menos tempo.
B) Ela serve para medir a estabilidade e confiabilidade do software, identificando a frequência de defeitos encontrados e ajudando a equipe a concentrar esforços nas áreas mais problemáticas.
C) A taxa de defeitos é utilizada principalmente para determinar o orçamento necessário para as fases de teste e manutenção do software.
D) Essa métrica é aplicada exclusivamente no final do ciclo de desenvolvimento para avaliar a eficácia das equipes de teste.
E) A principal utilização da taxa de defeitos é para comparar a qualidade do software com produtos concorrentes no mercado.

Questão 19 - No contexto do design orientado a componentes, como a gestão de dependências entre componentes afeta a manutenibilidade e a escalabilidade de uma aplicação de software?

A) A gestão de dependências é irrelevante em design orientado a componentes, pois todos os componentes são totalmente independentes.
B) Uma gestão de dependências eficaz minimiza o acoplamento entre componentes, facilitando atualizações, manutenção e escalabilidade da aplicação.
C) A gestão de dependências tende a aumentar o acoplamento entre componentes, resultando em uma manutenção mais complexa e reduzindo a escalabilidade.
D) Ela exige que cada componente compartilhe a mesma base de dados e recursos, aumentando a interdependência e complexidade da aplicação.
E) A gestão de dependências em design orientado a componentes foca exclusivamente na melhoria da performance, sem impactar a manutenibilidade ou escalabilidade.

Questão 20 - No desenvolvimento de software, como o princípio de "Inversão de Dependências" (Dependency Inversion) e a injeção de dependência contribuem para a criação de sistemas mais flexíveis e testáveis?

A) Esse princípio e técnica permitem que todas as dependências sejam centralizadas, facilitando a gestão e a atualização do código.
B) Eles garantem que o software possa operar sem qualquer dependência, tornando-o completamente autossuficiente e isolado.
C) A Inversão de Dependências e a injeção de dependência criam sistemas em que módulos de alto nível não dependem diretamente de módulos de baixo nível, mas de abstrações, tornando o sistema mais flexível e facilitando testes.
D) Estes conceitos são utilizados para duplicar todas as dependências, garantindo redundância e segurança no caso de falhas em um dos componentes.
E) A principal função é acelerar o processo de desenvolvimento, focando menos na qualidade do código e mais na entrega rápida.

Boa sorte!