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 - A Engenharia de Software surgiu como uma resposta a uma crise enfrentada pela indústria de software nas décadas de 1960 e 1970. Essa crise, conhecida como "Crise do Software", foi caracterizada por:

A) O excesso de investimentos governamentais em hardware e a escassez de profissionais de hardware.
B) O surgimento das metodologias ágeis, que tornaram os processos muito complexos.
C) A grande dificuldade em projetar circuitos integrados para computação de alto desempenho.
D) A incapacidade de entregar softwares no prazo, dentro do orçamento e com qualidade aceitável.
E) A rápida evolução da internet, que exigiu softwares mais simples e genéricos.

Questão 02 - Considere o seguinte cenário:
Uma equipe de desenvolvimento trabalha em um sistema para uma empresa de médio porte. Os requisitos são inicialmente bem compreendidos, mas os stakeholders pedem entregas frequentes de partes funcionais do sistema, permitindo feedback contínuo e ajustes ao longo do tempo. A equipe quer manter uma abordagem iterativa e incremental, com entregas funcionando a cada duas ou três semanas.

Diante desse cenário, o modelo de processo mais adequado para conduzir esse projeto é:

A) Modelo em Cascata
B) Modelo Espiral
C) Modelo Incremental com práticas ágeis, como Scrum
D) Modelo de Prototipação Evolutiva
E) Modelo V

Questão 03 - Considere um projeto de software que utiliza Git como sistema de controle de versão. Durante o desenvolvimento de uma nova funcionalidade, um desenvolvedor criou uma branch específica para essa tarefa. Após finalizar o trabalho, ele deseja incorporar suas alterações à branch principal (main) sem perder o histórico das contribuições anteriores.

Qual das estratégias abaixo é mais adequada para esse cenário?

A) Usar o comando git reset para remover o histórico da branch principal.
B) Usar git clone para duplicar o repositório com a nova funcionalidade.
C) Usar git merge para integrar a branch da funcionalidade à branch principal.

Boa sorte!

D) Usar git pull com a opção --force para forçar a sincronização com o repositório remoto.
E) Usar git revert para apagar a branch da funcionalidade antes de juntá-la à main.

Questão 04 - Considere as seguintes afirmações sobre qualidade de software, com base nos critérios propostos pela norma ISO/IEC 25010:

I. Funcionalidade, confiabilidade, usabilidade, eficiência de desempenho, manutenibilidade e portabilidade são características de qualidade de produto.
II. A segurança da informação e a compatibilidade entre sistemas são consideradas subcaracterísticas da confiabilidade.
III. A capacidade de o software operar com outros produtos de software é um aspecto relacionado à compatibilidade.
IV. A facilidade de identificar e corrigir falhas no código está associada à manutenibilidade.
V. A estética da interface e a satisfação do usuário fazem parte da subcaracterística de usabilidade.

Quais afirmações estão corretas?
A) Apenas I, III e V
B) Apenas II, III e IV
C) Apenas I, III, IV e V
D) Apenas I e IV
E) Todas as afirmações estão corretas

Questão 05 - O conceito de Integração Contínua (CI – Continuous Integration) em um ambiente DevOps refere-se a:
A) Automatizar testes de desempenho em ambientes de produção para validar escalabilidade.
B) Implementar firewalls para proteger pipelines de integração contra invasões externas.
C) Combinar mudanças de código frequentemente em um repositório compartilhado, com validação automática.
D) Criar ambientes distintos para desenvolvedores e testadores, garantindo isolamento entre versões.
E) Planejar sprints de desenvolvimento com antecedência mínima de três meses.

Questão 06 - Ao desenvolver uma API RESTful seguindo uma abordagem API First, qual das práticas abaixo é mais aderente aos princípios desse modelo?
A) Definir primeiro os testes de integração antes de escrever a documentação da API.
B) Projetar a interface da API, documentá-la com OpenAPI (Swagger), e só então iniciar a implementação.
C) Implementar o backend completo e depois gerar a especificação automática da API.
D) Codificar a API e permitir que os clientes descubram suas funcionalidades diretamente pelo código.
E) Começar a codificação com foco apenas nos requisitos não funcionais, deixando a interface para o final."

Questão 07 - Durante o desenvolvimento de software, os testes são classificados de acordo com o seu objetivo e momento de aplicação. Um teste de unidade tem como principal foco:
A) Verificar a integração entre diferentes subsistemas e módulos.
B) Validar se o software atende aos requisitos definidos pelos usuários finais.
C) Avaliar o comportamento do sistema sob alta carga e uso simultâneo.
D) Testar individualmente cada função ou método de forma isolada.
E) Executar casos de teste a partir de interfaces gráficas simuladas."

Questão 08 - Considere a seguinte situação:

Uma equipe está desenvolvendo um sistema de monitoramento de sensores industriais. Os dados chegam em alta frequência e precisam ser processados em tempo real. A

solução deve permitir escalabilidade horizontal, ser tolerante a falhas e facilitar a substituição de componentes individuais sem afetar o restante do sistema.

Com base nesse cenário, qual estilo arquitetural é o mais adequado?

A) Monolítico
B) Cliente-Servidor
C) Arquitetura em Camadas
D) Microsserviços
E) Orientado a Objetos

Questão 09 - Ao utilizar histórias de usuário (user stories) como forma de especificar requisitos em ambientes ágeis, espera-se que essas histórias:

A) Sejam descritas exclusivamente por analistas de sistemas e não sofram alterações após aprovadas.
B) Contenham todos os detalhes técnicos e diagramas UML necessários para implementação.
C) Foquem na perspectiva do usuário, com linguagem natural e estrutura simples, como ""Como [persona], quero [ação], para [benefício]"".
D) Sejam substituídas por casos de uso quando o projeto adotar metodologias ágeis.
E) Apresentem requisitos funcionais e não funcionais em formato padronizado e sem ambiguidade.

Questão 10 - No contexto do framework Scrum, qual das alternativas abaixo descreve corretamente o papel do Product Owner (PO)?

A) Coordenar os testes automatizados e gerenciar o pipeline de CI/CD.
B) Atuar como facilitador do time, removendo impedimentos e liderando as cerimônias Scrum.
C) Representar os interesses dos stakeholders e ser responsável por manter e priorizar o backlog do produto.
D) Codificar as funcionalidades mais críticas e revisar os commits antes de cada release.
E) Redigir histórias de usuário tecnicamente detalhadas e definir a arquitetura do sistema.

Questão 11 - O C4 Model é uma abordagem moderna para representar a arquitetura de software. Sobre essa abordagem, assinale a alternativa correta:

A) O nível de código-fonte (Code) representa a visão de alto nível de sistemas e suas dependências externas.
B) O nível de contêineres (Container) mostra como o sistema se comporta em tempo de execução e identifica gargalos de desempenho.
C) O nível de componentes (Component) detalha a estrutura interna de cada módulo em diagramas UML, com foco em algoritmos e controle de fluxo.
D) O nível de contexto (Context) descreve o sistema no seu ambiente, incluindo usuários e sistemas externos que interagem com ele.
E) O C4 Model não é compatível com sistemas distribuídos e orientados a serviços.

Questão 12 - Considere uma equipe que adota integração contínua (CI) e entrega contínua (CD) com testes automatizados. O time percebe que, apesar dos testes passarem no pipeline, falhas funcionais são detectadas em produção. Qual das alternativas melhor explica essa situação?

A) O pipeline de CI/CD está falhando por causa da ausência de um controle de versão adequado para o ambiente de produção.
B) A automação de testes foi mal implementada, utilizando apenas testes de carga e ignorando testes de unidade.
C) O time provavelmente está utilizando apenas testes de unidade, sem cobertura suficiente de testes de integração e testes end-to-end.
D) O problema está no uso de metodologias ágeis, que priorizam velocidade em detrimento da qualidade.
E) O ambiente de produção está desatualizado em relação ao ambiente de desenvolvimento, o que invalida a CI.

Boa sorte!

Questão 13 - Durante o ciclo de vida de um sistema, diferentes tipos de manutenção são aplicados para garantir sua continuidade e evolução. Qual das alternativas abaixo descreve corretamente a manutenção adaptativa?

A) Corrigir defeitos e falhas identificadas durante o uso do sistema em produção.
B) Adicionar novos recursos com base em mudanças nas necessidades dos usuários.
C) Alterar o software para que ele continue operando corretamente em um ambiente de execução modificado.
D) Reescrever partes do sistema para melhorar a estrutura interna, sem alterar seu comportamento externo.
E) Documentar tecnicamente os requisitos e funcionalidades do sistema para facilitar testes futuros.

Questão 14 - O paradigma do Desenvolvimento de Software Dirigido a Dados (DDSD) propõe que os dados sejam elementos centrais no ciclo de vida do software. Dentre as opções abaixo, qual prática mais fortemente caracteriza esse paradigma?

A) Adotar métodos ágeis com foco na entrega contínua de funcionalidades priorizadas pelo cliente.
B) Projetar APIs RESTful com base em princípios de design centrados no usuário final.
C) Construir sistemas baseados em regras de negócio predefinidas e documentação formal extensiva.
D) Utilizar fluxos de feedback contínuo e métricas derivadas de dados para tomada de decisão e melhoria incremental do software.
E) Implementar arquiteturas monolíticas que concentrem os dados em um único banco centralizado e relacional.

Questão 15 - No contexto de testes automatizados integrados ao ciclo de DevOps, qual das práticas abaixo mais contribui para a efetividade de testes contínuos (continuous testing) em um pipeline de integração e entrega contínua (CI/CD)?

A) Executar todos os testes manuais antes de cada build automático.
B) Concentrar os testes de regressão apenas ao final do desenvolvimento, antes do deploy.
C) Priorizar a execução de testes unitários durante o processo de design da arquitetura do sistema.
D) Integrar testes automatizados em diferentes níveis (unidade, integração, E2E) ao pipeline, com execução frequente e feedback rápido.
E) Adiar a execução dos testes automatizados para após a liberação do sistema em produção, usando monitoramento passivo.

Questão 16 - Uma equipe que adota práticas de DevSecOps deseja integrar segurança de forma contínua ao longo do ciclo de vida do software. Qual das práticas abaixo é mais alinhada aos princípios de DevSecOps?

A) Executar uma auditoria de segurança apenas no final do projeto, após a homologação.
B) Centralizar todas as decisões de segurança em um único especialista de segurança da informação.
C) Utilizar ferramentas de análise estática de código e scanners de vulnerabilidade integrados ao pipeline de CI/CD.
D) Separar o desenvolvimento da segurança para evitar conflitos de interesse durante o ciclo de desenvolvimento.
E) Confiar unicamente em firewalls e criptografia na camada de infraestrutura para proteger o sistema.

Questão 17 - Sobre os modelos de processo de software, considere as seguintes afirmações:

I. O modelo Cascata é adequado quando os requisitos estão bem definidos e estáveis desde o início do projeto.
II. O modelo Incremental permite entregas parciais do sistema em etapas, facilitando o feedback contínuo.
III. O modelo Espiral combina elementos iterativos com análise de riscos em cada ciclo.
IV. O modelo Ágil valoriza planejamento detalhado e documentação completa antes da codificação.

Boa sorte!

V. O modelo Híbrido busca adaptar práticas de diferentes modelos conforme as necessidades do projeto.

Assinale a alternativa correta:

A) Apenas as afirmações I, II e IV estão corretas.
B) Apenas as afirmações I, III e V estão corretas.
C) Apenas as afirmações II, III e IV estão corretas.
D) Apenas as afirmações I, II, III e V estão corretas.
E) Todas as afirmações estão corretas.

Questão 18 - Considere um cenário em que uma equipe de desenvolvimento de software implementou a Entrega Contínua (CD). Eles têm um pipeline de CD configurado para automatizar o lançamento de atualizações de software. No entanto, eles estão enfrentando um desafio comum: apesar de testes automatizados extensivos, algumas atualizações estão causando problemas pós-lançamento que não foram identificados durante os testes. Qual das seguintes estratégias deve ser priorizada para melhorar a eficácia do pipeline de CD neste cenário?

A) Diminuir a frequência dos lançamentos para permitir mais testes manuais.
B) Implementar um ambiente de 'staging' mais robusto que imite de perto o ambiente de produção.
C) Focar exclusivamente em aumentar a cobertura de testes automatizados.
D) Desativar temporariamente o pipeline de CD e voltar para a entrega manual.
E) Incentivar mais revisões de código por pares, independentemente dos testes automatizados.

Questão 19 - Em um ambiente de desenvolvimento de software ágil, como um Scrum Master pode efetivamente lidar com o desafio de membros da equipe trabalhando em ritmos significativamente diferentes, o que pode causar atrasos no projeto e afetar o moral da equipe?

A) Alocar os membros da equipe mais rápidos para as tarefas mais críticas e os mais lentos para tarefas menos prioritárias.
B) Implementar sessões de treinamento obrigatórias para os membros da equipe mais lentos, a fim de acelerar seu ritmo de trabalho.
C) Encorajar a equipe a estabelecer um ritmo de trabalho uniforme, pressionando os membros mais lentos a acompanhar os mais rápidos.
D) Facilitar a comunicação e a colaboração dentro da equipe para adaptar as cargas de trabalho e aproveitar as diferentes habilidades dos membros da equipe.
E) Mudar para um modelo de desenvolvimento de software tradicional, como o Waterfall, para melhor gerenciar as diferenças de ritmo.

Questão 20 - Como o padrão arquitetural "Microservices" influencia o processo de desenvolvimento e manutenção de aplicações de larga escala?

A) Microservices aumentam a dependência entre diferentes partes da aplicação, melhorando a eficiência do desenvolvimento.
B) Este padrão separa a aplicação em pequenos serviços independentes, o que pode aumentar a complexidade e a dificuldade de gerenciamento.
C) Microservices permitem o desenvolvimento, teste, implantação e escalabilidade de partes individuais da aplicação de forma independente, facilitando a manutenção e a evolução da aplicação.
D) A abordagem de microservices é ideal para aplicações pequenas e simples, devido à sua estrutura inerentemente complexa.
E) Microservices exigem que toda a aplicação seja reescrita a cada mudança, aumentando o tempo de desenvolvimento.

Boa sorte!