Exame de Engenharia de Software 2024/25 [90 min] PT

Época Normal

A fraude denota uma grave falta de ética e constitui um comportamento não admissível num estudante do ensino superior, revelando uma inaceitável falta de profissionalismo. Qualquer tentativa de fraude leva à anulação da prova tanto do facilitador como do prevaricador.
NOTA1: As respostas devem ser respondidas nesta folha de prova.
NOTA2: Preencha o seu nome e número de estudante no espaço correspondente abaixo.
NOTA3: Na sua zona de trabalho apenas deve estar esta folha, uma esferográfica e uma **página** A4 **manuscrita**.

Em todas as questões de escolha múltipla uma selecção errada envolve um desconto na cotação dessa questão inversamente proporcional ao número de opções corretas disponíveis (e.g. 1/4, 1/6, 2/8, 1/9).

Nome: _____ **Nr. de estudante:**_____

1. Quais as fases do ciclo de vida do software? (assinale todas as que se aplicam)

a) Levantamento de requisitos
b) Design arquitectural
c) Design detalhado
d) Construção e testes
e) Integração e testes
f) Entrega e disponibilização
g) Operações e manutenção
h) Conceptualização
i) Desactivação

2. Assinale as vantagens intrínsecas do modelo *Waterfall:*

a) Flexível às mudanças.
b) Robusto a alterações.
c) Baseado em riscos.
d) Cadência regular de entregas.
e) Conduz a boa documentação.
f) Fácil de planear.
g) Simples de entender.
h) Feedback regular pelo cliente.
i) Rapidez na execução.

3. Assinale aspectos de natureza externa que condicionam a opção por um modelo de software:

a) Dimensão do projeto
b) Política económica
c) Estabilidade do problema
d) Local (da equipa)
e) Orçamento do projeto
f) Faseamento dos pagamentos
g) Leis e regulamentos
h) Disponibilidade do cliente
i) *Open-source* ou proprietário

4. Em eng. de software o que se entende por 'gestão de configurações':

a) Conjunto de opções que define a forma como um programa funciona (e.g. *dark/light mode*)
b) Processo que determina como se fazem alterações ao código.
c) Práticas que determinam a utilização e formatos do repositório de projeto.
d) Conjunto de boas práticas que envolvem o *setup* da aplicação desenvolvida.

5. Que comando (git) armazena um *snapshot* do projeto num repositório gerido pelo git?

a) add
b) push
c) pull
d) switch
e) branch
f) snapshot
g) store
h) commit
i) revert

6A. Qual/quais das opções abaixo são mensagens de commit correctamente expressas:

a) `feature(api)!`
b) `feature(api)!: another commit`
c) `45:feature(api)!`
d) `45:feature(api)!: another commit`
e) `45:feature (api)`
f) `#45:feature (api)`
g) `#45: feature (api)`
h) `#45: feature (api): added read-repo`
i) `#45: feature (api)!: added read-repo`

7. O conceito de 'ramos de desenvolvimento' (*branching*) em git permite:

a) Que diferentes programadores possam trabalhar sem haver conflitos.
b) Que cada um dos membros da equipa possa trabalhar sem haver conflitos.
c) Que cada membro da equipa possa ter uma versão personalizada do repositório do projecto.
d) Que eventuais conflitos que possam ocorrer sejam perfeitamente identificados.
e) Que diferentes funcionalidades possam ser desenvolvidas no seu próprio ramo (apenas em *trunk-based*).

8. De acordo com o 'triângulo de ferro' de gestão de projeto de sofware, para garantir um produto de qualidade num projeto atrasado, qual o impacto de manter os custos e as funcionalidades?

a) Redução dos comentários. b) Redução dos custos. c) Sprints mais longas.
d) Má gestao dos riscos. e) Trabalho extra ao fim-de-semana. f) Funcionalidades incompletas.
g) Atraso na entrega. h) Ter de aumentar a equipa. i) Planeamento deficiente.

9. Os campos de uma tabela de *Burn-Down* incluem:

a) Linha de referência. b) Dias úteis da sprint. c) O membro da equipa
d) Items do backlog do produto. e) Estimativa inicial. f) Esforço remanescente
g) Entradas vazias para dias restantes. h) Gráfico com evolução do esforço. i) Esforço total estimado.

10. Um gráfico de Gantt inclui ou mostra:

a) As tarefas críticas(*) b) Se um projecto está atrasado. c) Percentagem de execução das tarefas
d) A duração das sprints. e) Uma lista de tarefas. f) As tarefas finalizadas.
g) A duração das tarefas. h) A data de entrega. i) O esforço aplicado em cada tarefa.

(*) aquelas cujo atraso impacta directamente a data de entrega final.´

11. Assinale o(s) riscos correctamente formulados:

a) O número de bugs está cada vez a aumentar mais.
b) Se não reduzirmos o numero de bugs o programa não vai estar pronto dentro do prazo.
c) O número de bugs está muito elevado, podemos não conseguir entregar a tempo.
d) O número de defeitos está muito elevado, podemos podemos não conseguir entregar dentro do prazo.
e) Não temos nenhum defeito identificado, pelo que corremos o risco de não ter defeitos no código.
f) Os testes de cobertura demonstram que há 10% de código por testar.

12. "*O tempo de execução dos testes de integração está a crescer todos os dias; a equipa pode começar a deixar de fazer merges para não ter de ficar à espera dos resultados*". Assinale possíveis estratégia(s) de <u>prevenção</u>:

a) Ir monitorizando para saber se a equipa está mesmo a deixar de fazer merges.
b) Com excepção dos merges para *main*, permitir que se continue o trabalho antes de finalizar um merge.
c) Aumentar a RAM dos servidores de integração para acelerar os testes.
d) Penalizar quem não fizer imediatamente merge quando uma funcionalidade estiver pronta.
e) Definir uma política de *selective testing*, em que apenas um subconjunto dos testes é executado em cada merge.
f) Alocar parte da equipa para escrever mais testes de aceitação e adquirir hardware adicional para os executar.

13. Em que contexto de colocam os "requisitos"?

a) Casos de uso b) User stories c) Funcionais
d) Não funcionais e) Condicionantes (restrições) f) Na qualidade.
g) No problema. h) Na solução. i) Na origem.

14. Quais destas actividades se classificam como de 'verificação'

a) Rever os requisitos. b) Aprovar os requisitos. c) Testes unitários.
d) Testes de Integração. e) Testes funcionais. f) Testes não funcionais.
g) Testes de aceitação. h) Testes de regressão. i) Testes de usabilidade.

15. Assinale os requisitos não-funcionais:

a) Nenhuma opção pode estar a mais de três écrans de distância da *landing page*.
b) O sistema tem de estar funcional durante as horas normais de expediente.
c) Inputs do utilizador final não nunca ser executados como código.
d) O utilizador deve poder apagar todos os dados da sua conta e removê-la.
e) A aplicação deve ter um interface audio para permitir a sua utilização por invisuais.
f) Nenhum componente de código pode ter uma complexidade ciclomática superior a 15.

16. Identifique vistas arquitecturais do modelo 4+1:

a) De sistema b) De aceitação c) Organigrama
d) Lógica e) De processo f) De software
g) De desenvolvimento h) De cenários i) Física

falta 1 **17. Identifique outros atributos de qualidade que <u>penalizam</u> o atributo 'usabilidade'**

a) Disponibilidade b) Eficiência c) Robustez
d) Testabilidade e) Portabilidade f) Manutenabilidade
g) Fiabilidade h) Flexibilidade i) Legalidade

→ **18. Qual dos padrões arquitecturais abaixo mais promove o atributo Escalabilidade (<u>só um</u>)**

a) Monolítico b) Cliente-servidor c) Em três camadas (e.g. MVC)
d) Pipe-and-Filter e) Microserviços f) Em quatro camadas.

→ **19. Qual das opções melhor descreve o *grau de interdependência* entre componentes de software?** (só uma)

a) A soma das complexidades dos métodos de uma classe. b) O número de sub-classes. c) O número de métodos de uma classe.
d) A coesão e) A sobreposição f) O acoplamento
g) A coincidência (dos métodos) h) A conexão i) A complexidade

20. Assinale os testes correctamente expressos:

a) Testa o login com uma password válida. b) Testa o login com uma password inválida.
c) Testa o login com user:"es" e pass: "2024" d) Testa o login com user:"es" e pass: "1234"
e) Testa o login com user:"es" e pass: "2024"; deve devolver erro. f) Testa o login com user:"es" e pass: "2024"; deve ir para a página 'home'

21. Na pirâmide de testes, quais os realizados <u>menos</u> frequentemente?

a) Unitários b) De regressão c) De validação
d) De aceitação e) De sistema f) De usabilidade
g) De verificação h) De integração i) Temporais

22. O que distingue um modelo de desenvolvimento <u>iterativo</u> de um modelo <u>incremental</u>? (escreva)

→ Descrição do modelo incremental
→ No entanto, é plan-based no qual não há mudança de reqs, ao contrário de modelos iterativos, no qual há várias iterações do produto (é um modelo do tipo ágil)