**Marque as respectivas respostas no gabarito abaixo. Cada questão vale 0,6 pontos.**

| **QUESTÃO:** | **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **Q6** | **Q7** | **Q8** | **Q9** | **Q10** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RESPOSTA**: | C | E | D | A | A | C | A | D | B | A |

1. **(Ano: 2024 Banca: IV - UFG).** Na engenharia de software, compreender as diferentes categorias de requisitos é fundamental para o sucesso do projeto. Considerando as diferentes categorias de requisitos, os requisitos funcionais são:
2. Condições ou capacidades que um sistema deve ter para ser operado.
3. Requisitos relacionados à segurança, à confiabilidade e ao desempenho do sistema.
4. Descrições do comportamento do sistema sob condições específicas.
5. Normas que regulam o desenvolvimento do sistema em um nível organizacional.
6. **(Ano: 2007 Banca: CESGRANRIO)**. O ciclo de vida clássico da engenharia de software tem sofrido críticas dos especialistas. Uma delas é o fato de o cliente ter de declarar todos os requisitos explicitamente na primeira parte do projeto, gerando insegurança. Uma técnica utilizada para minimizar esse problema é conhecida como:
7. Análise por ponto de função.
8. Casos de teste.
9. Linha básica ou baseline.
10. Projeto de software.
11. Prototipação.
12. **(Ano: 2010 Banca: BIO-RIO)**. No contexto da Engenharia de Software, segundo SUMMERVILLE, o processo de engenharia de requisitos inclui subsistemas de alto nível. Além da Viabilidade e Elicitação, compreende:
13. Análise e projeto;
14. Implementação e testes;
15. Especificação e validação;
16. Implantação e manutenção;
17. Projeto físico e projeto lógico.
18. **(Ano: 2024 Banca: Instituto Consulplan).** É o seu pior pesadelo. Um cliente entra em seu escritório, senta-se, olha diretamente nos seus olhos e diz: “Eu sei que você pensa que entendeu o que eu disse, mas o que você não entende é que aquilo que eu disse não era o que eu quis dizer”. Invariavelmente, isso acontece no final do projeto, após compromissos de prazos de entrega terem sido estabelecidos, reputações estarem em risco e muito dinheiro estar em jogo. Quem trabalhou na área de software e sistemas por mais do que alguns poucos anos já viveu esse pesadelo; mesmo assim, poucos aprenderam a livrar-se dele. Passamos por muitas dificuldades ao tentar extrair os requisitos de nossos clientes. Temos dificuldades para entender as informações obtidas. Normalmente, registramos os requisitos de forma desorganizada e investimos pouco tempo verificando aquilo que registramos. Deixamos que as mudanças nos controlem, em vez de estabelecermos mecanismos para controlar as mudanças. Em suma, não conseguimos estabelecer uma base sólida para o sistema ou software. Todos esses problemas são desafiadores. Quando combinados, o panorama é assustador até mesmo para os gerentes e profissionais mais experientes. Mas soluções existem. (PRESSMAN, 2021, p. 103. Adaptado.) Sobre a Engenharia de Requisitos, marque V para as afirmativas verdadeiras e F para as falsas.

( ) Recomenda-se iniciar a engenharia de requisitos desde o início de um projeto de software para garantir que o software atenda às necessidades do cliente.

( ) Durante a tarefa de elaboração na engenharia de requisitos, são identificados e analisados os cenários de usuário.

( ) A principal finalidade da negociação durante a engenharia de requisitos é eliminar completamente os requisitos conflitantes.

( ) A especificação de requisitos no contexto da engenharia de requisitos é um processo iterativo de captura e documentação dos requisitos.

( ) A rastreabilidade é importante na engenharia de requisitos para garantir que todos os requisitos estejam alinhados com as diretrizes de engenharia de software.

A sequência está correta em:

1. V, V, F, V, V.
2. V, F, V, F, V.
3. V, V, F, V, F.
4. V, F, V, F, F.
5. **(Ano: 2006 Banca: CEBRASPE)**. Com relação a engenharia de software, julgue os seguintes itens.
6. Já se tem livros repletos de padrões e procedimentos para elaborar software; isso fornece ao desenvolvedor tudo o que ele precisa.
7. A engenharia de software vai resultar na criação de documentação volumosa e desnecessária que certamente nos atrasará.
8. Se eu decidir terceirizar um projeto de software, vou poder relaxar e deixar que aquela firma o elabore.
9. Até que eu esteja com o programa executando, não tenho como avaliar a sua qualidade.
10. Os requisitos de projeto mudam continuamente, mas as mudanças podem ser facilmente acomodadas porque o software é flexível.
11. Possuímos ferramentas de desenvolvimento de software que estão no estado-da-arte, afinal compramos os computadores mais novos do mercado.
12. O único produto de trabalho que pode ser entregue para um projeto de software bem-sucedido é o programa executável.
13. Se o planejamento atrasar, é possível adicionar mais programadores e ficar em dia (conceito horda mongólica).
14. O estabelecimento geral de objetivos é suficiente para iniciar a escrita de programas; os detalhes podem ser fornecidos posteriormente.

Acerca da classificação desses itens, é correto afirmar que:

1. V e IX dizem respeito a mitos do cliente.
2. I, III, IV e IX dizem respeito a mitos da gerência.
3. II, IV e VI dizem respeito a mitos do profissional.
4. VI, VII e VIII dizem respeito a mitos de gerência.
5. **(Banca: Fundação Getúlio Vargas - FGV)**. No desenvolvimento centrado no usuário, tanto o conceito de MVP (Minimum Viable Product) quanto o de protótipo são essenciais. Acerca desse assunto, analise as afirmativas a seguir.
6. O protótipo permite testar ideias de produto no mercado real, reduzindo riscos financeiros ao evitar investimentos pesados antes de comprovar o interesse dos usuários.
7. O MVP deve apresentar o maior número possível de funcionalidades, de modo a avaliar corretamente o seu valor para o usuário.
8. Na engenharia de requisitos, os protótipos são empregados nas etapas de especificação, enquanto o MVP é empregado na validação.

Está correto o que se afirma em

1. I, apenas.
2. II, apenas.
3. III, apenas.
4. I e II, apenas.
5. II e III, apenas.
6. **(Ano: 2014 Banca: FUNCAB).** Em relação aos modelos de desenvolvimento de software, aquele que prioriza a análise dos riscos envolvidos no desenvolvimento de cada parte do software é:
7. Espiral.
8. De Prototipação
9. Cascata.
10. Desenvolvimento Baseado em Componentes
11. Incremental
12. **(Ano: 2024 Banca: Instituto Consulplan).** Relacione adequadamente os tipos de testes de software às suas respectivas descrições.

| 1. Unitário | ( ) Validar se o software é aceitável para uso de acordo com os requisitos e as necessidades de negócios. |
| --- | --- |
| 2. Integração | ( ) Garantir que essas partes funcionem bem juntas como um sistema coeso. |
| 3. Funcional | ( ) Verificar se o código-fonte de cada unidade funciona conforme o esperado. |
| 4. Aceitação | ( ) Certificar-se de que o software execute as ações esperadas e forneça os resultados corretos. |
| 5. Desempenho | ( ) Medir como o sistema se comporta sob diferentes condições de carga, identificando gargalos de desempenho. |

A sequência está correta em:

1. 1, 3, 2, 5, 4.
2. 4, 5, 3, 1, 2.
3. 3, 4, 2, 5, 1.
4. 4, 2, 1, 3, 5.
5. **(Ano: 2024 Banca: IBADE)**. Assinale a única alternativa que preenche corretamente a lacuna em branco. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_é um processo na Engenharia de Software responsável por identificar e controlar o acesso, as versões e as mudanças nos itens de configuração com o objetivo de garantir sua integridade.
6. Manutenção de Software.
7. Gerenciamento de Configuração.
8. Concurrent Versions System (CVS).
9. Subversion (CVN).
10. GIT.
11. **(Ano: 2024 Banca: FGV)**. Uma das ferramentas mais importantes no processo de desenvolvimento de software é o repositório de gestão de configuração de software utilizado. Ele oferece um conjunto de estruturas e mecanismos que auxiliam a equipe de desenvolvimento a gerenciar as modificações experimentadas por um software. Em termos de características, em um repositório de gestão de configuração é fundamental:
12. Acompanhar e manter as dependências entre os elementos de dados sob sua guarda.
13. Compatibilizar as estruturas de padrões de projetos GoF e GRASP ao longo do desenvolvimento do software.
14. Demandar um sistema de banco de dados externo para salvar documentos relacionados ao software.
15. Implantar métodos ágeis de levantamento de requisitos, como o QFD (*Quality Function Deployement*).
16. Reprogramar o software baseado no histórico de mudanças a partir da última versão salva.