|  |  |
| --- | --- |
|  | **Teste de Software** |
|  | Caderno de Exercícios |

|  |
| --- |
|  |
| Este documento contém os exercícios definidos para o curso Teste de Software. |

Abaixo estão alguns exercícios para amadurecer o que você aprendeu neste curso. Por favor, não tenha este exercício como “aqueles exercícios chatos de final de curso”. Eu não gosto de perder tempo ou encher meus alunos com teoria, então realmente dediquei muito tempo para, na sessão de respostas deste exercício, te dar explicações complementares importantes para que você amadureça, consolide o aprendizado e vá ficando mais familiarizado com o teste de software considerando aquilo que o mercado irá precisar de você. Vamos lá?!

1. **Quais atividades estão relacionadas com Teste?**
2. Atividades que envolvem a execução do teste sobre componente ou o sistema sob o teste.
3. Atividades somente no inicio do ciclo de vida do software.
4. Atividades do final do ciclo de vida do software.
5. Atividades durante todo o ciclo de vida do software.
6. Somente atividades que avaliam produtos de software.
7. **Qual é o propósito dos testes?**
8. Remover defeitos.
9. Minimizar os riscos causados por defeitos provenientes do processo de desenvolvimento.
10. Encontrar 100% dos defeitos do software.
11. Descobrir o maior número possível de defeitos do software, assegurar que o teste atende a todos os requisitos de sistema estabelecidos entre o desenvolvedor e o cliente.
12. **Qual(is) opção(ões) abaixo está(ão) incorreta(s)?**
13. Diferentes técnicas de teste são apropriadas para diferentes abordagens de engenharia de software e em diferentes pontos no tempo.
14. O teste é feito tanto pelo desenvolvedor do software como também por um grupo independente de teste.
15. Devemos nos preocupar com o teste somente após o código tiver sido gerado.
16. O teste começa no nível do componente e progride em direção à integração do sistema como um todo.
17. **Analise as questões abaixo e indique a mais correta:**
18. Desenvolvedores e testadores participam juntos de todas as etapas do desenvolvimento, pois é necessário para o sucesso do projeto.
19. Desenvolvedores e testadores integram equipes diferentes, pois a independência entre as equipes é necessária.
20. Desenvolvedores e testadores integram o mesmo time, mas com funções conflitantes.
21. Desenvolvedores e testadores são parte da mesma equipe, pois o trabalho de um depende do outro.
22. **É INCORRETO afirmar que:**
23. Um trabalho de qualidade depende também de um bom trabalho da equipe de testadores.
24. A revisão é importante durante todo o processo de desenvolvimento.
25. A qualidade de uma determinada tarefa depende do trabalho de todos os envolvidos.
26. Quando o programa a ser construído é pequeno, não justifica testar algo tão simples.
27. Ter desenvolvedores e testadores em equipes diferentes favorece a execução dos testes.
28. **É INCORRETO afirmar que:**
29. Os testes não garantem que o software ficará totalmente livre de defeitos.
30. Os testes diminuem consideravelmente os erros do software.
31. Os testes diminuem o risco de serem encontrados defeitos do software em produção.
32. Os testes, quando bem planejados e executados, eliminam 100% dos erros de um software.
33. É quase impossível testar todas as possibilidades de formas e alternativas de entradas de dados, ou todas as possibilidades e condições criadas pela lógica do programador.
34. **Quando o testador finaliza os testes que ele deveria fazer num determinado sistema, ele então:**
35. Corrige os erros de acordo com a prioridade e reexecuta os testes.
36. Procura o gerente do projeto e entrega os resultados, indicando que ele fez um ótimo trabalho.
37. Interrompe o projeto de desenvolvimento para falar sobre os resultados ruins, quando for o caso.
38. Dá seu parecer sobre o projeto de desenvolvimento, indicando se o trabalho foi ou não bem feito.
39. Entrega o resultado do seu trabalho ao responsável pelos testes (seu líder na equipe de testes).
40. **Os testes de caixa-branca são baseados nos elementos internos de um trecho de código. Quais níveis ou tipos de teste abaixo são caixa branca?**
41. Teste de Sistema
42. Teste de Desempenho
43. Análise Estática
44. Teste Alfa
45. Teste de Unidade
46. **Em que fase do desenvolvimento de um sistema as atividades abaixo são realizadas?**

a) Execução do Teste Alfa: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b) Execução dos Testes de Unidade: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

c) Criação dos cenários para o Teste de Sistema: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

d) Execução do Teste Beta: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Além de dominar os conhecimentos sobre teste de software, o que significa ver e rever este curso, além de buscar oportunidades de colocar em pratica na sua vida profissional, deixo (relembrando) aqui alguns conselhos para que você seja um testador de alto nível:

1. **Seja detalhista** e nunca deixe de testar determinada funcionalidade por achar que ela é simples demais e o programador “não deve ter cometido um erro ali”. É aí que você se engana, pois programadores possuem dias bons e maus, dias de muita vontade e dias de grande preguiça de revisar o código feito. Você, como um testador, deve desconfiar de TUDO!

2. **Considere cada detalhe do documento de especificação do sistema.** Quando você ler a documentação de um sistema com a intenção de testá-lo, faça isso com 200% de atenção e preciosismo. Teste cada palavra, cada afirmação descrita no documento sobre o que o sistema de ou não deve fazer. O ótimo testador não deixar de testar absolutamente nada do que foi especificado.

3. Aprenda a **ser neutro e se colocar no lugar do usuário** que irá utilizar o sistema após seu lançamento. Pense como ele, haja como ele. Não deixe seus “vícios” ou pré julgamentos influenciar no seu teste.

4. **Descreva os erros encontrados com o máximo de precisão e informação**. Isto ajudará a equipe de desenvolvimento a entender, analisar melhor o problema e resolvê-lo o mais rápido possível.

5. **Sinta orgulho quando lhe chamarem de chato ☺** Isso mesmo. Quando você ouvir um desenvolvedor dizer: “aquele testador é chato pra caramba! Pega no pé! Vai no detalhe!”, saiba que você está fazendo um excelente trabalho!!!

Um grande abraço e, por favor, mantenha-me informado sobre a sua experiência daqui pra frente com os testes de software, pode ser? Me adiciona nas redes sociais e vamos manter contato.

Grande abraço e, de todo coração, te desejo muito sucesso!!!

Gustavo Farias

|  |  |  |
| --- | --- | --- |
| Questão | Resposta | Comentários e Explicações |
| 1 | D | . As atividades de teste ocorrem ao longo do processo de desenvolvimento de software.  **. Antes da codificação do sistema:** lembre-se que, por exemplo, os cenários de ‘Teste de Aceitação’ e ‘Teste de Sistema’ são construídos antes da codificação do sistema ser realizada pelos programadores.  **. Durante a codificação do sistema:** temos os testes de unidade e integração, normalmente feitos pelos programadores.  **. Após a codificação do sistema:** temos a execução dos testes de sistema, aceitação, alfa e beta. |
| 2 | D | a) Testes **não** removem defeitos. Testes encontram defeitos.  c) **Nenhum** de teste, por mais longo e perfeito que seja, garante que 100% dos defeitos de um sistema serão encontrados. Lembra que as combinações de entradas e saídas são próximas do infinito? Esta é a principal barreira, pois você geralmente executa uma parcela muito pequena das possíveis ações que os usuários reais poderão realizar no sistema. |
| 3 | C e D | c) Como já falado anteriormente, o trabalho com os testes iniciam assim que o primeiro documento de especificação do sistema é aprovado, pois os cenários dos testes de aceitação já podem começar a ser construídos.  d) O teste começa no nível da unidade (ex.: uma classe ou um método) e não do componente. |
| 4 | B | b) Para que sejam criados testes eficientes, uma prática chave é que os cenários de teste sejam elaborados por pessoas que não estão participando da codificação do sistema, de modo a evitar “vícios” e “otimismos” na hora de criar os cenários que validarão o sistema.  . **Vícios** porque o desenvolvedor, acostumado a convivência com o sistema, pode acabar focando os testes apenas naquilo que considerou mais crítico para ele enquanto codificava. Já um testador externo está isento de influências e fará uma análise neutra do sistema para definir o que é mais ou menos crítico de ser testado.  . **Otimismos** porque, naturalmente, quando um programador codifica algo com a intenção de fazer o melhor, confia no que faz e, na hora de testar, testará relaxadamente, afinal, “já pensou com bastante calma enquanto estava codificando”. Assim, os testes realizados por programadores tendem a ser pouco exigentes e de baixa qualidade. |
| 5 | D | Todo e qualquer sistema, grande ou pequeno, complexo ou simples, precisa ser bem testado. Lembre-se que quanto mais tarde se encontra um problema, mais caro é para resolvê-lo. |
| 6 | D | **Nenhum** de teste, por mais longo e perfeito que seja, garante que 100% dos defeitos de um sistema serão encontrados. Lembra que as combinações de entradas e saídas são próximas do infinito? Esta é a principal barreira, pois você geralmente executa uma parcela muito pequena das possíveis ações que os usuários reais poderão realizar no sistema |
| 7 | E | Um testador não é responsável por nada além de entender a especificação do sistema (i.e. o que o sistema deve fazer) e testá-lo. Corrigir as falhas encontradas é responsabilidade da equipe de programadores e nunca da equipe de teste. |
| 8 | C e E | c) A análise estática é, por definição, caixa branca, uma vez que é composta por validações visuais realizadas sobre o código desenvolvido.  e) O teste de unidade é feito pelo próprio desenvolvedor e parte do conhecimento que ele tem sobre a estrutura do código que ele fez, tal como testes condicionais, loops e etc. Assim, é um teste que envolve uma análise prévia do código feito. |
| 9 | - | a) Após os testes de sistema terem sido concluídos e todos os erros encontrados corrigidos.  b) Ainda na fase de codificação cada desenvolvedor deve criar testes de unidade para validar o próprio código. Esta é a primeira validação realizada sobre um trecho de código concluído.  c) Pode ser iniciado assim que o documento de especificação do projeto for aprovado/liberado e pode ser concluído até antes da data planejada para o início dos testes de sistema.  d) O teste Beta (grupo maior de usuários testando a aplicação) tem início logo após a conclusão do teste Alfa (grupo menor de usuários testando a aplicação). |