**1** [**Q2066779**](https://www.qconcursos.com/questoes-de-concursos/questoes/3d92cdbf-a4)[Engenharia de Requisitos](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/engenharia-de-requisitos)

**Prova:**[FGV - 2023 - CGE-SC - Auditor do Estado - Ciências da Computação - Tarde (Conhecimentos Específicos)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-cge-sc-auditor-do-estado-ciencias-da-computacao-tarde-conhecimentos-especificos)

A respeito do requisito não funcional, assinale V para a afirmativa verdadeira e F para a falsa.

( ) O sistema deve permitir que os usuários criem e editem seus perfis. ( ) O sistema deve estar disponível na Web e em dispositivos Android. ( ) O sistema deve ter um recurso de pesquisa que permita aos usuários encontrar informações sobre vendas. ( ) O sistema deve estar disponível 99,99% do tempo. ( ) O sistema deve exibir dados de vendas em tempo real.

As afirmativas são, respectivamente,

**A** V – F – V – F – V.

**B** V – V – V – F – V.

**C** F – V – F – V – F.

**D** V – F – V – V – V.

**E** V – F – F – F – V.

Vegeta Concurseiro

08 de Fevereiro de 2023 às 23:23

**- Requisitos Funcionais**

- O que o sistema fornece e como o sistema reage a entradas especificas

- Funcionalidades do software

- Relacionadas a casos de uso

**- Requisitos Não-Funcionais**

- São restrições nos serviços ou funções oferecidas pelo sistema

- Afetam arquitetura do sistema

**2** [**Q2066777**](https://www.qconcursos.com/questoes-de-concursos/questoes/3d8eaed7-a4) [Processos de Software - Desenvolvimento Ágil](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/processos-de-software-desenvolvimento-agil)

**Prova:**[FGV - 2023 - CGE-SC - Auditor do Estado - Ciências da Computação - Tarde (Conhecimentos Específicos)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-cge-sc-auditor-do-estado-ciencias-da-computacao-tarde-conhecimentos-especificos)

Assinale a opção que, em metodologias ágeis, apresenta corretamente a principal diferença entre a "Definição de Pronto" (DoR) e a "Definição de Feito" (DoD).

**A** A Definição de Pronto define quando uma tarefa pode ser trabalhada, enquanto a Definição de Feito define quando uma tarefa é concluída.

**B** A Definição de Pronto define quando um recurso pode ser trabalhado, enquanto a Definição de Feito define quando um recurso é concluído

**C** A Definição de Pronto define os critérios de aceitação para uma tarefa, enquanto a Definição de Feito define os critérios de aceitação para uma história de usuário.

**D** A Definição de Pronto define critérios de aceitação para um recurso, enquanto a Definição de Feito define critérios de aceitação para um lançamento.

**E** A Definição de Pronto define quando uma história de usuário pode ser trabalhada, enquanto a Definição de Feito define quando uma história de usuário é concluída.

Plinio Passos

23 de Março de 2023 às 20:09

Errei..

DoR é aplicada antes do desenvolvimento do item, enquanto a DoD é aplicada após o desenvolvimento. A DoR é um conjunto de critérios para que um item de trabalho seja considerado pronto para ser desenvolvido, enquanto a DoD é um conjunto de critérios para que um item de trabalho seja considerado pronto para ser entregue ao cliente ou usuário final.

Luiz Felipe Neskovek Goulart

20 de Abril de 2023 às 18:47

Pronto: está pronto pra começar. Tudo definido para começar a desenvolver.

Feito: Está feito, pode entregar. Pode entregar pro cliente.

**3** [**Q2066776**](https://www.qconcursos.com/questoes-de-concursos/questoes/3d8ca25d-a4) [Processos de Software - Desenvolvimento Ágil ,](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/processos-de-software-desenvolvimento-agil)[Scrum](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/scrum)

**Prova:**[FGV - 2023 - CGE-SC - Auditor do Estado - Ciências da Computação - Tarde (Conhecimentos Específicos)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-cge-sc-auditor-do-estado-ciencias-da-computacao-tarde-conhecimentos-especificos)

No *Scrum*, o principal objetivo da reunião de Retrospectiva da Sprint é

**A** revisar e atualizar o *backlog*do produto.

**B** avaliar o progresso feito durante a *sprint*.

**C** planejar a próxima *sprint*.

**D** identificar e planejar melhorias para a próxima *sprint*.

**E** revisar e atualizar o *backlog* da *sprint*.

Vegeta Concurseiro

08 de Fevereiro de 2023 às 23:10

**Revisão da Sprint (Sprint Review)**

- É Proporcional a 4 horas

- É executada no final da sprint para inspecionar o incremento e adaptar o backlog do produto

- Inspeciona o resultado da Sprint e determinar as adaptações futuras.

**Retrospectiva da Sprint (Sprint Retrospective)**

- É Proporcional a 3 horas

- é uma chance para o Scrum Team inspecionar a si próprio e criar um plano de melhorias para a próxima sprint.

- Planejar medidas que possam trazer, no próximo Sprint, melhorias relacionadas à colaboração entre as pessoas, processos ou ferramentas.

**4** [**Q2066775**](https://www.qconcursos.com/questoes-de-concursos/questoes/3d8a76dc-a4) [Teste de Software](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/teste-de-software)

**Prova:**[FGV - 2023 - CGE-SC - Auditor do Estado - Ciências da Computação - Tarde (Conhecimentos Específicos)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-cge-sc-auditor-do-estado-ciencias-da-computacao-tarde-conhecimentos-especificos)

O tipo de teste de *software* que serve para garantir que todas ou algumas partes de um sistema estão dialogando e funcionando corretamente em conjunto é o teste

**A** de regressão.

**B** de aceitação.

**C** de integração.

**D** de validação.

**E** unitário.

Vegeta Concurseiro

08 de Fevereiro de 2023 às 23:13

**Teste de integração**

* Tratam de projeto de software
* Envolve a construção de um sistema com base em seus componentes
* Verifica se seus componente funcionam realmente em conjunto
* Descobre erros de interface
* É um teste caixa-cinza
* Problema principal é a localização do erro
* Integração TOP-DOWN = Sistemas depois componentes
* Integração BOTTOM-UP = Componente depois Funcionalidades

Rian Rosal

13 de Março de 2023 às 17:32

Nesse teste, em vez de se atestar funcionalidades do software, **se analisa a integração entre as diferentes unidades que formam o sistema.** São averiguados aspectos como a**interface e a dependência** entre os componentes.

confere se todas (ou algumas) partes estão dialogando e funcionando corretamente em conjunto.

joao danielak

02 de Abril de 2023 às 19:18

Palavra pra Teste de integração, desta pergunta, foi dialogando e funcionando, pois este teste faz funcionamento com todos ou quase todos os módulos de um sistema

**5** [**Q2031071**](https://www.qconcursos.com/questoes-de-concursos/questoes/1b5007a1-92) [Orientação a Objetos](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/orientacao-a-objetos)

**Prova:**[FGV - 2023 - SEFAZ-MG - Auditor Fiscal da Receita Estadual - Tecnologia da Informação (Tarde)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-sefaz-mg-auditor-fiscal-da-receita-estadual-tecnologia-da-informacao-tarde)

Os padrões de projeto de software (*design patterns*) tiram proveito máximo dos pilares da orientação a objetos. Usemos como exemplo o padrão *Abstract Factory*, que é um padrão de projeto criacional que permite produzir famílias de objetos relacionados sem especificar suas classes concretas.

O principal pilar da orientação a objetos usado nesse design pattern é

**A** o polimorfismo.

**B** a herança.

**C** o encapsulamento.

**D** a abstração.

**E** a sublevação.

Thaís Santos

18 de Janeiro de 2023 às 11:52

**GAB.: LETRA 'A'**

Vamos por partes.

“Abstract Factory (…) é um padrão criacional que permite produzir famílias de objetos relacionados (…)”

O que seria “famílias de objetos relacionados”?

Imagine uma classe que seja responsável por criar uma cadeira, uma mesa de centro e um sofá. Esses são itens utilizados em uma sala de estar. Então, essa classe é responsável por criar “famílias de objetos relacionados”.

“(…) sem especificar suas classes concretas.”

Agora, imagine a seguinte situação: existem diversos tipos desse conjunto (cadeira, mesa de centro, sofá), como o estilo moderno, antigo, casa de campo.

Digamos que eu queira criar um conjunto do tipo “caso de campo”, eu vou precisar criar uma classe do zero ou posso aproveitar a base daquele conjunto que criei? Posso aproveitar a base, afinal toda cadeira tem 4 pernas, por exemplo. Não preciso implementar isso toda hora.

Dessa forma, se eu quero criar um conjunto no estilo moderno, pego a base e modifico o que for necessário, para dar esse toque moderno. Mesmo para o estilo antigo e casa de campo.

Essa ideia de “pego a base e modifico de acordo com a minha necessidade” se refere à qual característica da Orientação a Objetos?

Polimorfismo: “(…) **princípio a partir do qual as classes derivadas de uma única classe base são capazes de invocar os métodos que, embora apresentem a mesma assinatura, comportam-se de maneira diferente para cada uma das classes derivadas**.”

Fontes: https://refactoring.guru/pt-br/design-patterns/abstract-factory

https://www.devmedia.com.br/conceitos-e-exemplos-polimorfismo-programacao-orientada-a-objetos/18701

Eder Luiz da Silva

03 de Março de 2023 às 15:05

**Gabarito: Letra A.**

**O padrão Abstract Factory utiliza polimorfismo para permitir a criação de objetos relacionados sem especificar suas classes concretas.**

**Esse padrão define uma interface abstrata que declara um conjunto de métodos para criar objetos relacionados, e várias classes concretas que implementam essa interface para criar diferentes famílias de objetos.**

**Ao invocar os métodos dessa interface abstrata, o código cliente pode criar objetos de diferentes famílias sem conhecer as classes concretas desses objetos. Isso é possível porque o polimorfismo permite que objetos de diferentes classes possam ser tratados de forma polimórfica, ou seja, através de um tipo comum que define um conjunto de métodos que esses objetos devem suportar.**

**Dessa forma, o polimorfismo é fundamental no padrão Abstract Factory para permitir que o código cliente possa trabalhar com diferentes famílias de objetos sem conhecer suas classes concretas, o que torna o código mais flexível e extensível.**

Ana Beatriz Pereira de Souza

13 de Março de 2023 às 16:09

O padrão Abstract Factory usa o polimorfismo para produzir famílias de objetos sem especificar suas classes

Os 4 pilares da orientação a objetos

**POLIMORFISMO** = produz famílias de objetos sem especificar suas classes

**HERANÇA**= herda atributos da sua superclasse

**ABSTRAÇÃO**= é o primeiro pilar, é implementado através de classes, que contém propriedades e métodos, de forma bastante simples.

**ENCAPSULAMENTO**= é realizado através de propriedades privadas, auxiliadas por métodos especiais getters e setters, Getters e setters são usados ​​para proteger seus dados, principalmente ao criar classes. Para cada variável de instância, um método getter retorna seu valor enquanto um método setter define ou atualiza seu valor.

**6** [**Q2031070**](https://www.qconcursos.com/questoes-de-concursos/questoes/1b4db61e-92)[Processos de Software - Desenvolvimento Ágil](https://www.qconcursos.com/questoes-de-concursos/disciplinas/tecnologia-da-informacao-engenharia-de-software/processos-de-software-desenvolvimento-agil)

**Prova:**[FGV - 2023 - SEFAZ-MG - Auditor Fiscal da Receita Estadual - Tecnologia da Informação (Tarde)](https://www.qconcursos.com/questoes-de-concursos/provas/fgv-2023-sefaz-mg-auditor-fiscal-da-receita-estadual-tecnologia-da-informacao-tarde)

Em uma equipe ágil, um dos papéis mais importantes é o do responsável por planejar o desenvolvimento do produto, escolhendo e priorizando os itens do *backlog* e garantindo que o máximo de valor seja entregue a cada *sprint*.

Assinale a opção que indica o nome correto desse membro do time.

**A** Scrum Master

**B** Stakeholder

**C** Gerente de Projetos

**D** Product Owner

**E** Desenvolvedor

Bruno Caldara

24 de Janeiro de 2023 às 10:13

O **Product Owner** tem as seguintes responsabilidades:

* Definir as funcionalidades do produto ();
* Priorizar as funcionalidades de acordo com o valor de negócio;
* Ajustar funcionalidades e prioridades a cada Sprint, conforme necessário;
* Garantir que o  seja visível, transparente e claro para todos;
* Garantir que o  entenda os itens do  no nível necessário;
* Decidir a data de liberação e conteúdo do Release; e,
* Aceitar ou Rejeitar os resultados de trabalho.

Fonte: https://www.trt9.jus.br/pds/Scrum/roles/product\_owner\_10E7BD3.html

marilia moreira

20 de Abril de 2023 às 10:45

famigerado "P.O."