

**TECNÓLOGO SISTEMAS PARA INTERNET**

**Professor**: Heitor José dos Santos Barros

**Resumão – Programação Orientada a Objetos**

**Data de disponibilização do trabalho no Moodle, pelo professor**: 06/12/2020

**Data de entrega da tarefa pronta, pelo aluno**: 27/12/2020

**Valor**: 2,00

**Atenção**: Trabalhos entregues fora da data só serão aceitos mediante apresentação de atestamento médico.

**Identificação**

**Nome: JOAO PAULO PIRES DANTAS**

Para cada um dos conceitos, procure 2 autores que os contextualizam, informando o livro que foi retirado a citação e a referida página.

**1 – POO - Programação Orientada a Objetos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **8** | **Java – Como Programar** | **[...] Objetos ou, mais precisamente, as classes de onde objetos vêm são essencialmente componentes reutilizáveis de software. Há objetos data, objetos data/hora, objetos áudio, objetos vídeo, objetos automóvel, objetos pessoas, etc. Quase qualquer substantivo pode ser razoavelmente representado como um objeto de software em termos dos atributos (por exemplo, nome, cor e tamanho) e comportamentos (por exemplo, calcular mover e comunicar). Grupos de desenvolvimento de software podem usar uma abordagem modular de projeto e implementação orientados a objetos para que sejam muito mais produtivo do que com as técnicas populares como “programa estruturada” – programas orientados a objetos são muitas vezes mais fáceis de entender, corrigir e modificar.** |
| **Thiago Leite e Carvalho** | **2017** | **5** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **O Paradigma Orientado a Objeto (POO) tem como principal característica uma melhor e maior expressividade das necessidades do nosso dia a dia. [...] possibilita criar unidades de código mais próximas de transformação das necessidades diárias para uma linguagem orientandas a objetos.** |

**2 – Herança**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **8** | **Java – Como Programar** | **Uma nova classe de objetos pode ser criada convenientemente por meio de herança – ela (chamada subclasse) começa com as características de uma classe existente (chamada superclasse), possivelmente personalizando-as e adicionando aspectos próprios. Na nossa analogia do carro, uma objeto da classe “conversível” decerto é um objeto mais geral “automóvel”, mas especificamente, o teto pode ser levantado ou baixado.** |
| **Thiago Leite e Carvalho** | **2017** | **30** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **Na herança, é possível criar classes a partir de outras classes. Como consequência disto, ocorre um reaproveitamento de códigos – dados e comportamentos – da chamada classe mãe. Neste caso, a classe filha, além do que já foi reaproveitada, pode acrescentar o que for necessário para si.** |

**3 - Encapsulamento**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **8** | **Java – Como Programar** | **Classes (e seus objetos) encapsulam, isto é, contêm seus atributos e métodos. Os atributos e métodos de uma classe (e de seu objeto) estão intimamente relacionados. Os objetos** |
| **Thiago Leite e Carvalho** | **2017** | **32** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **[...] A complexidade que desejamos esconder é a implementação de alguma necessidade. Com o encapsulamento, podemos esconder a forma como algo foi feito, dando a quem precisa apenas o resultado gerado. Não importa para quem requisitou algum processamento/comportamento ter conhecimento da lógica realizar para gerar o resultado final. Apenas o resultado obtido é que é relevante.** |

**4 – Polimorfismo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Thiago Leite e Carvalho** | **2017** | **113** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **Em determinados momentos em uma hierarquia de classes, precisamos que um mesmo método (nome e lista de parâmetro, ou seja, assinatura) se comporte de forma diferente dependendo do objeto instanciado a partir de uma classe de uma hierarquia qualquer. Isto surge devido à necessidade de flexibilidade que a hierarquia de classe deseja fornecer. [...] A grande vantagem do uso do polimorfismo é que podemos utilizar objetos distintos e continuar executando a mesma ação, sendo que esta ação se moldará ao objeto corrente.** |
| **Paul Deitel e Harvey Deitel** | **2016** | **312** | **Java – Como Programar** | **O polimorfismo permite “programas no geral” em vez de “programar” no específico. Em particular, o polimorfismo permite escrever programas que processam objetos que compartilham a mesma superclasse, direta ou indiretamente, como se todos fossem objetos da superclasse, isso pode simplificar a programação.** |

**5 – Polimorfismo de sobreposição**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **290** | **Java – Como Programar** | **[...] usa a anotação @Override opcional para indicar que a seguinte declaração (isto é, toString) deve sobrescrever um método da superclasse existente. Essa anotação ajuda o compilador a capturar alguns erros comuns. [...] Quando o compilador encontra um método declarado com @Override, ele compara a assinatura desse método com as assinaturas dos métodos da superclasse. Se não houver correspondência exata, o compilador emite uma mensagem de erro [...]** |
| **Victorio Albani de Carvalho e Giovany Frossard Teixeira** | **2012** | **60** | **Programação Orientada a Objetos** | **[...] A notação @Override é inserida aumaticamente pelo NetBeans para indicar que esse método foi definido no ancestral e está sendo redefinido na classe atual.** |

**6 – Polimorfismo de sobrecarga**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Thiago Leite e Carvalho** | **2017** | **55** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **Muitas vezes, é preciso que um mesmo método possua entradas (parâmetros) diferentes. Isso ocorre porque ele pode precisar realizar operações diferentes em determinado contexto. Este processo é chamado de sobrecarga de método.  Para realiza-la, devemos manter o nome do método intacto, mas alterar sua lista de parâmetros. Podem ser acrescentados ou retirados parâmetros para assim se prover um novo comportamento.** |
| **Guilherme Silveira e Mário Amaral** | **2017** | **231** | **Java SE 8 Programmer I – O gui para sua certificação Oracle Certified Associate** | **Um método pode ter o mesmo nome que outro, desde que a sua invocação não fique ambígua: os argumentos que são recebidos têm de ser obrigatoriamente diferentes, sejam quantidade ou em tipos[...].** |

**7 - Interface**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **10** | **Java – Como Programar** | **O Java também suporta interfaces – coleções de métodos relacionados que normalmente permitem informar aos objetos o que fazer, mas não como fazer. [...]**  **Uma classe implementa zero ou mais interfaces – cada uma das quais pode ter um ou mais métodos -, assim como um carro implementa interfaces separadas para as funções básicas de dirigir, controlar o rádio, controlar os sistemas de aquecimento, ar-condicionado e afins.** |
| **Thiago Leite e Carvalho** | **2017** | **132** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **Em algumas aplicações orientadas a objetos que necessitam de uma modelagem um pouco mais elaborada, muitas vezes é preciso determinar um conjunto de métodos que devem obrigatoriamente ser usados. Porém, como esses métodos serão realmente implementados, não importa a quem definiu tal conjunto. Essa obrigatoriedade de definição de métodos é chamado de interface.** |

**8 – Coleções de objetos**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **538** | **Java – Como Programar** | **Uma coleção é uma estrutura de dados – na realidade, um objeto – que pode armazenar referências a outros objetos. Normalmente, coleções contêm referências a objetos de qualquer tipo que tem o relacionamento é um com o tipo armazenado na coleção. As interfaces de estrutura de coleções declaram as operações a ser realizadas genericamente em vários tipos de coleções.** |
| **Thiago Leite e Carvalho** | **2017** | **208** | **Orientação a Objetos – Aprenda seus conceitos e suas aplicabilidades de forma efetiva** | **É pensando em tornar mais simples (alto nível) o uso dessas estruturas que linguagens orientadas a objetos possuem classes especialmente criadas para facilitar a manipulação delas: as coleções. Tanto Java como C# possuem coleções.** |

**9 – Tratamento de exceções**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **348** | **Java – Como Programar** | **[...] uma exceção é uma indicação de um problema que ocorre durante a execução de um programa. O tratamento de exceção permite criar aplicativos que podem resolver (ou tratar) exceções. Em muitos casos, o tratamento de uma exceção permite que um programa continua executando como se nenhum problema tivesse sido encontrado.** |
| **Guilherme Silveira e Mário Amaral** | **2017** | **358** | **Java SE 8 Programmer I – O gui para sua certificação Oracle Certified Associate** | **Durante a execução de uma aplicação, erros podem acontecer. A linguagem Java oferece um mecanismo para que o programador possa definir as providências apropriadas a serem tomadas na hora em que o erro de execução ocorrer. Esse mecanismo nos permite controlar o que deve acontecer quando tais situações surgirem.  Os erros de execução são classificados em algumas categorias. [...] A classificação das categorias depende exclusivamente da hierarquia das classes que modelam os erros de execução como objetos em Java.** |

**10 – MVC – Model View Controller**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Autor(es)** | **Ano** | **Pagina** | **Nome do Livro/Artigo** | **Citação** |
| **Paul Deitel e Harvey Deitel** | **2016** | **874** | **Java – Como Programar** | **[...] Seguem o padrão de design Model-View-Controller (MVC), que separa os dados de um aplicativo (contidos no modelo), de um lado, e, de outro, a GUI do aplicativo (a visualização) e a lógica de processamento do aplicativo (o controlador.**  **O controlador implementa a lógica para processar entradas do usuário. O modelo contém dados do aplicativo e a visualização apresenta os dados armazenadas no modelo. Quando um usuário fornece alguma entrada, o controlador modificar o modelo com a entrada dada. [...] Quando o modelo muda, o controlador atualiza a visualização para apresentar os dados alterados.** |
| **Paulo Silveira, Guilherme Silveira, Sérgio Lopes, Guilherme Moreira, Nico Steppat e Fabio Kung** | **2017** | **206** | **Introdução à arquitetura e design de software – uma visão sobre a plataforma JAVA** | **O modelo MVC vem justamente para facilitar a separação do layer visual da aplicação do domínio e do negócio. Na definição formal de MVC (BURBECK, 1987), a atualização da view é feita através de observers que são notificados dos eventos ocorridos nos modelos, uma visão muito ligada às aplicações desenvolvidos na época para Desktop. Mas, apesar da definição formal desse padrão ou das interpretações que o mercado adotou do MVC, a característica de maior valor é a de separar os componentes de apresentação do resto da aplicação (conhecimento como Separated Presentation)(FOWLER, 2003a).** |