Centro Universitário UNA
Programação de Soluções Computacionais
Professor Daniel Henrique Matos de Paiva

Lista de Revisão VI

Responda as questões abaixo:

1. Contextualização: Em **Programação Orientada a Objetos**, a visibilidade de uma variável é determinada pelo seu nível de acesso. Uma variável privada só pode ser acessada dentro da classe em que foi declarada, o que garante a encapsulação dos dados e a segurança do programa.
   Nesse contexto, vamos abordar questões sobre variáveis privadas.
   Questão: Considere uma classe chamada "Pessoa" que possui uma variável privada chamada "idade". Essa variável armazena a idade de uma pessoa e só pode ser acessada dentro da classe. Com base nessa informação, qual das alternativas abaixo é verdadeira?
   a) Variáveis privadas não podem ser utilizadas em nenhuma situação.
   b) Variáveis privadas podem ser acessadas diretamente por qualquer método da classe.
   c) Variáveis privadas podem ser acessadas por métodos públicos da classe.
   d) Variáveis privadas podem ser acessadas por qualquer classe do programa.
   e) Variáveis privadas só podem ser acessadas por métodos privados da classe.

2. Levando em conta o paradigma de **Programação Orientada a Objetos**, é correto afirmar que podemos utilizar um relacionamento entre os objetos que permite a derivação de classes utilizando processo de especialização ou generalização dos dados e métodos com o objetivo de criar subclasses ou superclasses. O nome do relacionamento citado é:
   a) Agregação
   b) Associação
   c) Herança
   d) Composição
   e) Encapsulamento

3. **Herança** é um princípio de orientação a objetos, que permite que classes compartilhem atributos e métodos, conforme nos apresenta Cai Horstmann em sua obra "Conceitos de Computação Java. Na programação orientada a objetos, qual das seguintes afirmações sobre herança está correta?
   Referência:
   HORSTMANN, Cay. Conceitos de Computação com Java. São Paulo: Grupo A, 2009. E-book, p. 393. ISBN 9788577804078. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788577804078/. Acesso em: 07 nov. 2023.
   a) Uma classe filha pode herdar construtores da classe pai
   b) Uma classe pai herda automaticamente todos os métodos da classe filha
   c) A herança só pode ocorrer entre classes que estão no mesmo pacote
   d) Uma classe filha não pode acessar os membros protegidos da classe pai
   e) A classe filha somente tem acesso direto aos atributos privados da classe pai

4. Dentro das linguagens de programação, há algumas formas diferentes de se escrever um laço de repetição. Uma dessas formas é denominada "**for-each loop**", conforme apresenta Herbert Schildt em sua obra "Java para Iniciantes". Embora possamos produzir o mesmo resultado com cada um deles, alguns apresentam certa vantagem para determinados cenários. Qual é o propósito do "for-each" loop (for-each enhanced loop)?
   Referência:
   SCHILDT, Herbert. Java para Iniciantes. P. 150. São Paulo: Grupo A, 2015. E-book. ISBN 9788582603376. Disponível em: https://integrada.minhabiblioteca.com.br/#/books/9788582603376/. Acesso em: 07 nov. 2023.
   a) Realizar iterações sobre todos os elementos de um array ou coleção
   b) Controlar o fluxo de execução em uma instrução "for"
   c) Substituir a necessidade de usar a instrução "if" em um loop "for"
   d) Executar operações assíncronas em um ambiente de thread único
   e) Tem a mesma função de um laço duplo, ou seja, encadeado

5. Considere que você está desenvolvendo um sistema de gerenciamento de biblioteca. Nesse sistema, você tem uma classe base chamada Livro, que contém atributos como titulo, autor, isbn e métodos para gerenciar esses atributos. Para diversificar as funcionalidades do sistema, você decide criar novas classes para diferentes tipos de livros, como LivroFisico e Ebook, que terão atributos específicos adicionais, como peso para LivroFisico e formatoArquivo para Ebook, além de compartilhar os atributos da classe Livro. Nesse cenário, qual é o papel da **herança** na programação orientada a objetos?
   a) Permitir a criação das classes LivroFisico e Ebook com base na classe Livro, estendendo suas funcionalidades e reutilizando código para gerenciar atributos comuns como titulo, autor, e isbn.
   b) Garantir que todas as classes, incluindo Livro, LivroFisico, e Ebook, compartilhem exatamente os mesmos métodos e atributos, para promover uma padronização completa do código em todo o sistema de gerenciamento da biblioteca.
   c) Fornecer uma maneira de desenvolver interfaces de usuário mais complexas especificamente para LivroFisico e Ebook, facilitando a interação dos usuários com diferentes tipos de livros no sistema operacional.
   d) Encapsular dados dentro de métodos privados na classe Livro, assegurando que o estado interno dos objetos LivroFisico e Ebook só possa ser alterado através de operações definidas na classe Livro.
   e) Utilizar a herança para criar um sistema de controle de versões para os livros, permitindo que LivroFisico e Ebook representem diferentes versões do mesmo título, mantendo um histórico de alterações.