Programação Orientada à Objetos

Questão 01

Tratando-se de programação orientada a objeto, em Java, uma **classe** é um espaço padrão onde os objetos serão criados. Por sua vez, **objeto** se trata de um membro, dotado de identidade, comportamento e estado particulares, pertencente a uma classe.

Assim como os objetos, dentro de classes, ou não, comumente são instanciados **métodos**, que são blocos de código com ações e comportamentos específicos que serão executados quando o(s) bloco(s) for(em) solicitado(s) em algum momento.

Instanciado dentro das classes, os atributos referem-se ao comportamento das características de determinado objeto dentro dos métodos. Quanto à privacidade dessas características, há o **encapsulamento** que trata-se do escopo de visibilidade dos atributos e métodos, sobre o quanto aquilo pode ser exibido ou mantido sob proteção. Assim, a visibilidade de atributos e métodos é diferenciada por três designações. São elas **public, private, protected**, sugestivamente, a visibilidade/acesso dos atributos e/ou métodos é regida por três tipos: public, cuja visibilidade/acesso é livre; private, com acesso restrito à classe; e protected, cuja relação de acesso é possível às classes de um mesmo pacote.

Outra relação importante na programação orientada a objeto é a **herança.** Essa relação é caracterizada por estender às classes “filhas” todas as características da classe pai. Ainda sob a perspectiva de herança, **polimorfismo** é uma característica que sintetiza que classe-pai e classe-filho têm as mesmas possibilidades. Já dizia Dr. Fischer: “onde passa o pai, o filho passa”. Além disso, há as **classes** do tipo **abstrato**. Essas classes surgem tão somente para atuarem como classes-pai e, assim, serem herdadas.

Para instanciar uma classe se faz necessário criar um **construtor**. Por vezes, os atributos estão sob um escopo que não permite sua visibilidade em determinada classe, assim são criados os métodos **get** e **set** para permitir a captação daquele atributo e usá-lo em determinada classe e/ou método. É nesse momento que usa-se outros artifícios para fazer as devidas referências. Para tal existem as palavras reservadas **super**, **this** e **final**. “Super” faz referência à classe pai; “this” serve para referenciar a própria classe; “final” ter classes herdeiras.

Sobre métodos, há dois eventos a serem diferenciados, são eles a **sobrecarga de métodos** e a **sobrescrita de métodos**. Sobrecarregar um método significa ter vários métodos com nome iguais, mas assinaturas diferentes, e sobrescrever um método, ou propriedade, significa dar uma nova forma a eles, uma nova versão.

Por fim, na programação orientada a objetos, a relação entre classes geram herança e as classes herdeiras é condicionada pelas referências **ter**, **usar** e **ser.** Assim, as três designações condicionam a relação entre classes e objetos, além de tratar da composição e associação destes.