1. Na programação orientada a objetos, há um mecanismo que permite definir modificadores de acesso. Quando se define um atributo de uma classe com o modificador de acesso privado, significa que
   1. o acesso à classe é privado.
   2. o atributo é acessível a um programa que tenha uma referência a um objeto da classe.
   3. a classe é abstrata.
   4. o atributo é acessível somente aos métodos da classe.
2. A respeito de construção de sistemas, julgue o item subsequente.

Em orientação a objetos, os membros de dados de uma classe devem ser acessados por um método específico do objeto, e não diretamente.

* 1. Certo
  2. Errado

1. Julgue o item seguinte, a respeito de programação orientada a objetos.

O tipo de herança mais eficiente e indicado é a herança de implementação, pois possibilita que uma nova classe reutilize a implementação de outra classe sem a necessidade de se recortar e colar o código de forma manual, tornando o código automaticamente disponível, como parte da nova classe.

* 1. Certo
  2. Errado

1. Uma classe abstrata A contém o método abstrato foo(), que não foi reimplementado pela classe B que herda de A. Nesse contexto,
   1. a criação de uma classe abstrata C, que herda de B, requer a implementação do método foo().
   2. a implementação do método foo() em B é obrigatória para que ela compile.
   3. a chamada do método foo() de um objeto de B chamará a implementação existente em A.
   4. d a classe B não pode sobrecarregar o método foo().
2. A programação orientada a objetos:
   1. impossibilita o polimorfismo sem herança (extends em JAVA).
   2. impede a construção de software de difícil manutenção.
   3. faz uso de conceitos como classe, interfaces e envio de mensagens.
   4. requer que classes sejam estendidas para a reutilização de código.
3. O paradigma de programação Orientada a Objetos − OO utiliza, como um de seus componentes essenciais, a classe. Uma classe, em conformidade com os melhores padrões da OO,
   1. deve ter alta coesão, que implica em ter um conjunto limitado de responsabilidades, e baixo acoplamento, que implica em ter baixa dependência de outros componentes.
   2. deve ser completa, portanto, quanto mais atributos os métodos da classe tiver em comum com outros métodos, mais completa ela se torna.
   3. deve manter o número de colaborações com outras classes, por meio de seus objetos, o mais alto possível para facilitar os testes.
   4. que possui muitos métodos contribui para diminuir a complexidade da árvore de herança, aumentando seu potencial de reutilização.
   5. que possui métodos é chamada de superclasse e uma classe que não possui métodos é chamada de subclasse.
4. Com relação aos conceitos de orientação objeto, existe uma característica que faz com que detalhes internos do funcionamento dos métodos de uma classe permaneçam ocultos para os objetos e que por conta dessa técnica, o conhecimento a respeito da implementação interna da classe é desnecessário do ponto de vista do objeto, uma vez que isso passa a ser responsabilidade dos métodos internos da classe. A característica apresentada se refere a:
   1. encapsulamento.
   2. polimorfismo.
   3. abstração.
   4. herança.
   5. Namespaces.
5. Na programação orientada a objetos são utilizados classes e métodos. Um método
   1. com determinado nome não pode aparecer mais de uma vez na mesma classe, mesmo que receba parâmetros de tipos diferentes.
   2. em uma superclasse pode ser sobrescrito nas subclasses de uma relação de herança.
   3. em Java pode receber diversos parâmetros e ter diversas operações de retorno consecutivas de tipos diferentes.
   4. deve ter assinatura e corpo quando aparecer em uma interface Java e deve ser público.
   5. construtor em uma classe Java pode ser sobrecarregado, desde que tenha um tipo de retorno diferente de void.
6. Na Programação Orientada a Objetos, a alteração do funcionamento interno de um método herdado de um objeto pai consiste em:
   1. encapsulamento.
   2. abstração.
   3. herança.
   4. Polimorfismo.
7. Assinale a opção que apresenta o mecanismo a partir do qual os detalhes da implementação dos métodos de uma classe são ocultos dos usuários da classe.
   1. herança múltipla
   2. especialização
   3. herança simples
   4. classe abstrata
   5. encapsulamento