Programação Orientada a Objetos – Python
Prof. Diógenes Furlan
Revisão do 2º Bimestre

---

1) No paradigma orientado a objetos, as classes são abstrações de elementos do mundo real. Os dados de uma classe não podem e não devem ser manipulados diretamente por uma funcionalidade implementada em outra classe.

   Na implementação de uma classe, qualquer alteração nos dados de uma classe deve acontecer pela invocação de um método da própria classe. Esta proteção é conhecida como:

   a) Encapsulamento
   b) Abstração
   c) Atributo
   d) Método
   e) Procedimento

2) No âmbito dos princípios de concepção e programação orientada a objeto, é correto afirmar que "um objeto da subclasse é, também, um objeto da superclasse, ou seja, os objetos da subclasse podem ser tratados como objetos da superclasse".

   Esta afirmação é possível quando se refere ao contexto de:

   a) Abstração
   b) Polimorfismo
   c) Herança
   d) Encapsulamento
   e) Reutilização

3) **Em POO (Programação Orientada a Objetos), dizer que a classe A estende a classe B é o mesmo que dizer que:**

   a) as classes A e B são irmãs.
   b) a classe B é subclasse de A;
   c) a classe B é derivada de A;
   d) a classe A é superclasse de B;
   e) a classe A é derivada de B;

4) Os acessos e alterações dos dados de um objeto acontecem por meio de métodos implementados nesse objeto, para evitar que ocorram acessos diretos aos dados e assim evitando erros de alterações. Por esta característica, os dados ficam escondidos para dentro do objeto.

   Tal característica é conhecida como:

   a) Abstração
   b) Polimorfismo
   c) Herança Múltipla
   d) Encapsulamento
   e) Dependência

5) Os objetos de uma classe podem herdar atributos e métodos de mais de uma classe base. Pode introduzir complexidade, como o problema do diamante de herança e ambiguidades.

   Tal característica é conhecida como:

   a) Abstração
   b) Polimorfismo
   c) Herança Múltipla
   d) Encapsulamento
   e) Dependência

6) É a característica única de linguagens orientadas a objetos que permite que diferentes objetos respondam a mesma mensagem cada um a sua maneira. Em termos de programação, representa a capacidade de uma única referência invocar métodos diferentes, dependendo do seu conteúdo.

   Tal característica é conhecida como:

   a) Abstração
   b) Polimorfismo
   c) Herança Múltipla
   d) Encapsulamento
   e) Dependência

7) Sejam A e B duas classes em um programa orientado a objetos. Se A é _____ de B, então objetos da classe A _____ atributos que objetos da classe B.

Assinale a alternativa que completa correta e sequencialmente as lacunas do texto.

a) superclasse; possuem necessariamente menos.
b) superclasse; possuem necessariamente mais.
c) subclasse; possuem necessariamente menos.
d) subclasse; podem possuir mais.
e) subclasse; não podem possuir mais.

8) O reaproveitamento do código na programação orientada a objeto é um dos principais benefícios do uso desse paradigma, analise o código a seguir, escrito na linguagem Java:

```
public class Blusa {
    public void cor ( );
}

class Camiseta extends Blusa {
}

class Camisa extends Blusa {
    public void cor ( );
}
```

Sobre reaproveitamento, analise as seguintes afirmativas:

I. a classe Camisa implementa um outro método cor, diferente daquele da classe Blusa.
II. a palavra extends na classe Camiseta define a relação de interface entre Camisa e Blusa.
III. é possível observar o conceito de classe abstrata nessas classes.
IV. a classe Camisa poderá fazer uso de métodos pela herança direta da classe Blusa.
V. a classe Blusa, sendo privada, pode ser acessada em qualquer lugar do projeto.

Está correto o que se afirma:

a) Apenas nas afirmativas I e II
b) Apenas nas afirmativas I e IV
c) Apenas nas afirmativas I e II e IV
d) Apenas nas afirmativas II e IV
e) Nas afirmativas I, II e V

9) O código a seguir foi escrito utilizado a linguagem C#. Analise as classes nele escritas.

```
1       class A
2       {
3           public double x { get; set; }
4
5           public A(double x)
6           {
7               this.x = x;
8           }
9           public virtual void calcula(int a, int b)
10          {
11              x = x + a + b;
12          }
13          public virtual void calcula(double a, double b)
14          {
15              x = x + a – b;
16          }
17    }
18  class B:A
19  {
20    public double z {get; set; }
21
22    public B(int x, int z): base(x)
23    {
24          this.z = z;
25    }
26    public B(double x, double z): base(x)
27    {
28          this.z = x;
29    }
30
31    public override void calcula(double a, double b)
32    {
33      base.calcula(a,b);
34      z = x + z;
35    }
36  static void Main()
37  {
38      A obj1 = new A(5);
39      B obj2 = new B(3, 4.5);
40      obj1.calcula(1,2);
41      obj2.calcula(2.5, 1.5);
42      double resposta = obj1.x + obj2.x + obj2.z;
43      MessageBox.Show(resposta.ToString());
44  }
45
46
```

De acordo com o código analisado, considere as seguintes asserções:

I. o encapsulamento pode ser visto na linha 42;
II. na linha 18 a classe B está herdando as características da classe base A;
III. a linha 26 contém polimorfismo (Sobrecarga)
IV. na linha 38 temos instanciação de classe em memória stack;

Está correto o que se afirma:

a) Somente na afirmativa I
b) Apenas na afirmativa IV
c) Apenas nas afirmativas I e II
d) Apenas nas afirmativas II e III
e) Nas afirmativas I, II e III

10) Verifique o código a seguir e selecione quais conceitos de orientação a objetos estão sendo utilizados:

```
1  <?php
2  class ChequeComum
3  {
4          public $valor;
5
6          public function setValor( $valor )
7          {
8                  $this->valor = $valor;
9          }
10         public function getValor()
11         {
12                 return $this->valor;
13         }
14         public function calculaJuros()
15         {
16             return $this->valor * 1.25;
17         }
18 }
19 ?>
```

```
1  <?php
2  class ChequeEspecial extends ChequeComum
3  {
4          public function calculaJuros()
5          {
6             return $this->valor * 1.10;
7          }
8  }
9  ?>
```

Os conceitos de orientação a objetos que estão sendo utilizados são definidos como:

I. polimorfismo e herança.
II. herança e encapsulamento
III. encapsulamento e abstração.
IV. herança, encapsulamento e polimorfismo.

Está correto o que se afirma:

a) Somente na afirmativa I
b) Apenas na afirmativa III
c) Apenas nas afirmativas I e II
d) Apenas nas afirmativas II e III
e) Nas afirmativas I, II e III

Parte Prática

1) Fazer uma classe abstrata PlacaDeVideo com os atributos
    - Marca
    - Modelo
    - Qtd Memória
    - Clock

    E os métodos:
    - Construtor
    - Informação (mostrar)
    - Calcular performance (Qtd_memória x Clock)

2) Fazer um classe PlacaDeJogos (herança de PlacaDeVideo), com mais o atributo
    - Suporte ray tracing (sim/não)

   E os métodos:
    - Construtor
    - Informação (mostrar)

3) Fazer um classe PlacaProfissional (herança de PlacaDeVideo), com mais o atributo
    - Certificações de software (lista de string)

   E os métodos:
    - Construtor
    - Informação (mostrar)

4) Fazer uma classe Coleções de PlacaDeVideo com um atributo
    - Lista de placa (lista)

   E os métodos:
    - Construtor
    - Adicionar placa
    - Mostrar modelo placa com melhor performance

5) No main, criar 2 objetos de cada classe e inserir na coleção. Então rodar os métodos:
    - Informação (para cada objeto)
    - Calcular performance (para cada objeto)
    - Melhor performance