A+ Alterar modo de visualização

Peso da Avaliação 1,50
Prova 48787945
Qtd. de Questões 10
Acertos/Erros 9/1
Nota 9,00

1 O encapsulamento é um dos pilares da programação orientada a objetos, e sua utilização de forma correta serve como embasamento para os outros dois pilares: a herança e o polimorfismo. Com relação ao encapsulamento, assinale a alternativa CORRETA:

A Para fazer uso do encapsulamento, basta modificarmos a visibilidade de nossos atributos.

B O encapsulamento permite que os objetos se comuniquem através de mais caminhos.

C Através do encapsulamento, podemos proteger os valores internos dos objetos de acessos desnecessários.

D O objetivo principal do encapsulamento é obter um alto acoplamento entre as classes.

2 Programação é o processo de escrita, teste e manutenção de um programa de computador. A Programação Orientada a Objetos (POO) diz respeito a um padrão de desenvolvimento que é seguido por muitas linguagens, como C# e Java. Com relação aos conceitos de programação orientada a objetos e sua implementação na linguagem Java, analise as afirmativas a seguir:

I- Classes abstratas servem como modelo para outras classes que dela herdam e devem ser instanciadas utilizando a palavra reservada "new".
II- Todas as classes em Java herdam, direta ou indiretamente, a classe Object.
III- Interfaces podem ser utilizadas quando classes diferentes (não relacionadas) precisam de funcionalidades comuns (métodos) ou utilizam constantes comuns.
IV- Na linguagem Java, é possível haver polimorfismo por meio da implementação de herança entre classes.

Agora, assinale a alternativa CORRETA:

A As afirmativas I e III estão corretas.

B Somente a afirmativa II está correta.

C As afirmativas II, III e IV estão corretas.

D As afirmativas I e IV estão corretas.

3 O paradigma da orientação a objetos aplicada no desenvolvimento de software possui a vantagem de tornar o código-fonte reutilizável, mais legível, e fácil de realizar manutenção. O polimorfismo permite que outras classes representem o comportamento de classes que elas

referenciam. Com relação às classes criadas no código-fonte, assinale a alternativa CORRETA:

```
(1)public class Veiculo {
(2)    public void parar(){}
(3)}
(4)
(5)public class Motocicleta extends Veiculo{
(6)    public void parar(){
(7)        System.out.println("Parando a motocicleta!!");
(8)    }
(9)}
(10)
(11)//Veiculo utilitario esporte (SUV: Sport Utility Vehicle)
(12)public class Suv extends Veiculo{
(13)    public void parar(){
(14)        System.out.println("Parando o veículo utilitário!!");
(15)    }
(16)}
(17)
(18)public class Principal {
(19)    public static void main(String[] args){
(20)        Veiculo veiculo1 = new Motocicleta();
(21)        Veiculo veiculo2 = new Suv();
(22)
(23)        veiculo1.parar();
(24)        veiculo2.parar();
(25)    }
(26)}
```

A As classes Motocicleta e Suv referenciam o comportamento da classe Veiculo, e ao executarmos a classe Principal, será impresso primeiramente a mensagem (Parando o veículo utilitário!!) e depois (Parando a motocicleta!!).

B Ao executarmos a classe Principal, ocorrerá um erro designando que a instância da classe Veiculo nas linhas 20 e 21 são inválidas, pois recebem a instância de classes de outro tipo (Motocicleta e Suv).

C A classe Veiculo referencia o comportamento das classes Motocicleta e Suv, e ao executarmos a classe Principal, será impresso primeiramente a mensagem (Parando a motocicleta!!) e depois (Parando o veículo utilitário!!).

D Não existe polimorfismo implementado entre as classes do código-fonte apresentado.

4 O conceito de classes abstrata, através de suas características especiais, fornece ao conceito de herança uma maior confiabilidade, uma vez que as classes modeladas com herança precisam ser melhor idealizadas e projetadas. Dessa forma, avalie o código-fonte da classe "Atleta":

```
package fontes;

public class Atleta{
   private int pais;
   private String modalidade;
   private boolean medalistaOlimpico;

   public abstract void competir(String modalidade) {
      System.out.println("É especialista em " + modalidade);
```

```
    }

    public void competirPor(int pais){
        System.out.println("O atleta compete por " + pais);
    }

}
```

I- A classe em questão não apresenta nenhum problema de sintaxe.
II- O método "competirPor(int pais)" está gerando um erro de sintaxe, e adicionando a palavra-chave "abstract", antes da palavra chave "void", resolverá o erro gerado.
III- O método "competirPor(int pais)" não gera erros de sintaxe, logo não necessita da palavra-chave "abstract", antes da palavra-chave "void".
IV- O método "competir(String modalidade)" gera erro de sintaxe.

Assinale a alternativa CORRETA:

A As sentenças III e IV estão corretas.

B As sentenças II e III estão corretas.

C As sentenças I e II estão corretas.

D As sentenças II e IV estão corretas.

5

Existem três características mais importantes na Programação Orientada a Objetos (POO). O Conceito diz que uma operação pode ser definida em mais de uma classe (hierarquicamente correlacionadas), podendo assumir diferentes implementações, em cada uma dessas classes.

Assinale a alternativa CORRETA com o termo que é completamente definido pela assertiva:

A Herança.

B Polimorfismo.

C Associação unária.

D Sobrecarga de operadores.

6

A própria API do Java implementa a herança em diversos locais. Nas APIs descritas a seguir, assinale a alternativa INCORRETA:

A MessageFormat, ChoiceFormat e SimpleDateFormat.

B TextFormat, MessageFormat e ChoiceFormat.

C Format, NumberFormat e SimpleDateFormat.

D NumberFormat, DateFormat e DecimalFormat.

7

A Sobrecarga ou overloading é usada para implementar métodos que realizam tarefas similares para argumentos de tipos diferentes ou ainda para quantidades diferentes de argumentos. Sobre de uma função do Java que utiliza o recurso de sobrecarga, analise as sentenças a seguir:

I – Função printf(). Você pode passar uma mensagem ou uma soma. printf(1+1).

II – Math.pow(x, y)

III – Função garbage collector.

IV – Package Sobrecarga.

V – public class.

Assinale a alternativa CORRETA:

A As sentenças I e III estão corretas.

B As sentenças I e IV estão corretas.

C As sentenças I e II estão corretas.

D As sentenças II e V estão corretas.

8 Os modificadores de acesso são palavras reservadas da linguagem de programação Java e definem os padrões de visibilidade de acesso às classes, aos atributos e aos métodos. Por intermédio dos modificadores de acesso, é possível ocultar determinadas partes do código, dividindo o programa em partes menores e independentes e esta ação, de ocultar e/ou dividir, é conhecida como encapsulamento Acerca do conceito de encapsulamento e dos modificadores de acesso da linguagem de programação Java, analise as afirmativas a seguir:

I- Uma declaração com o modificador "public" permite o acesso de qualquer lugar do projeto Java e por qualquer entidade que possa visualizar a classe a que ela pertença.
II- Os membros de uma classe declarados com o modificador "protected" não podem ser acessados ou utilizados por nenhuma outra classe.
III- Métodos declarados como "default" só podem ser acessados a partir dos métodos da própria classe.
IV- O Java define quatro modificadores de visibilidade: "private", "public", "protected" e "default".

Agora, assinale a alternativa CORRETA:

A Somente a afirmativa I está correta.

B As afirmativas I e IV estão corretas.

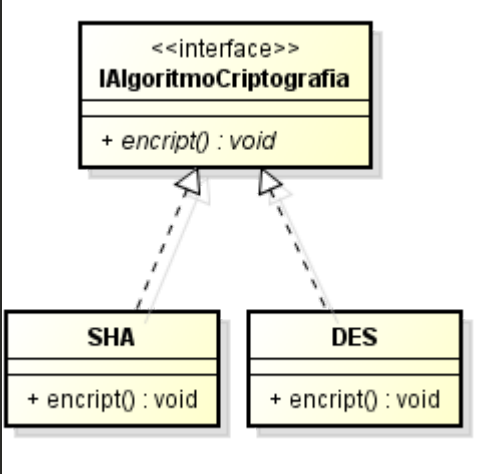
C As afirmativas II e III estão corretas.

D As afirmativas II, III e IV estão corretas.

9 O Polimorfismo de inclusão permite que se defina um novo comportamento para um objeto sem que se faça alterações nos que já estão funcionando, minimizando consideravelmente os custos e reduzindo o tempo para novas implementações. Considerando que a figura a seguir traz duas classes relacionadas a uma interface, classifique V para as sentenças verdadeiras e F para as falsas:

( ) O método encript() da interface é implicitamente abstrato.
( ) Na linguagem de programação Java, demonstraríamos que SHA implementa IAlgoritmoCriptografia através da palavra reservada extends.
( ) Um método existente em outra classe cuja assinatura seja “public void criptografar(IAlgoritmoCriptografia iac)” pode receber como parâmetro tanto um objeto da classe SHA quanto um objeto da classe DES.
( ) Podemos dizer que tanto SHA quanto DES implementam IAlgoritmoCriptografia.

Agora, assinale a alternativa que apresenta a sequência CORRETA:



A V - V - F - V.

B V - V - F - F.

C V - F - V - V.

D F - V - F - V.

10

O conceito de sobrecarga permite que você utilize o mesmo nome de método para muitos métodos diferentes, cada um com um número e tipos de parâmetros distintos. A sobrecarga é útil quando um método não é definido por seus argumentos e sim um conceito independente dos parâmetros. Sobre o conceito de sobrecarga, analise as sentenças a seguir:

I – É considerado um tipo polimorfismo.

II – Para que a sobrecarga aconteça, basta que se diferencie o número de parâmetros ou o tipo de parâmetros.

III – É considerado um tipo de Abstração.

IV – É considerado um tipo de Herança.

V – É considerado um tipo de encapsulamento.

Assinale a alternativa CORRETA:

A As sentenças II e V estão corretas.

B As sentenças I e II estão corretas.

C As sentenças I, III e IV estão corretas.

D As sentençasa I e III estão corretas.

Imprimir