UNIVERSIDADE DO OESTE DE SANTA CATARINA
Curso: Desenvolvimento Web, Cloud e Dispositivos Móveis
Componente curricular: Programação Orientada a Objetos
Professor: André Luiz Forchesatto

Avaliação

Aluno(a): Marcelo Garbin Data: 14/05/2016

Questões captadas de vários concursos públicos, disponíveis no portal http://www.questoesdeconcursos.com.br/

1.) (1 ponto) Assinale a alternativa correta sobre a orientação a objeto:

RESPOSTA: C

a) Herança permite o reaproveitamento de atributos e métodos, porém, isso não altera o tempo de desenvolvimento, não diminui o número de linhas de código e não facilita futuras manutenções.

b) Em uma aplicação que utiliza herança múltipla, uma superclasse deve herdar atributos e métodos de diversas subclasses. Todas as linguagens de programação orientadas a objeto permitem herança múltipla.

c) O polimorfismo associado à herança trabalha com a redeclaração de métodos previamente herdados por uma classe. Esses métodos, embora semelhantes, diferem de alguma forma da implementação utilizada na superclasse, sendo necessário, portanto, reimplementá-los na subclasse.

d) Em uma relação de herança é possível criar classes gerais, com características compartilhadas por muitas classes. Essas classes não podem possuir diferenças.

2.) (1 ponto) Polimorfismo é: (assinale a alternativa correta)

RESPOSTA: E

a) a multiplicidade de atributos de determinada classe.
b) a propriedade de um diagrama de classes ter múltiplas classes possuidoras de atributos.
c) a habilidade de um atributo ou variável poder identificar instâncias de classes com atributos dependentes.
d) a propriedade de uma instrução poder apontar para múltiplos objetos de uma mesma classe sem implicações de desempenho.
e) a habilidade pela qual uma única operação ou nome de atributo pode ser definido em mais de uma classe e assumir implementações diferentes em cada uma dessas classes.

3.) (1 ponto) Existem algumas maneiras de restringir o acesso a atributos e métodos, três delas são as mais conhecidas, cite quais são e qual a diferença entre elas.

RESPOSTA:
As três maneiras de restringir o acesso a atributos e métodos mais conhecidas são:
public: É utilizado em atributos e métodos universais, com isso os mesmos podem ser acessados de qualquer parte da aplicação.
private: É utilizado em atributos e métodos que necessitam de um controle maior na aplicação, com isso ao utilizar este modificador de acesso não é possível acessa-los através de subclasses e/ou em outras partes da aplicação a não ser na classe que foram criados.
protected: Com este modificador, apenas a classe que contém o modificador e os tipos derivados dessa classe tem o acesso aos seus atributos e métodos.

4.) (1 ponto) Com relação aos conceitos de programação orientada a objetos, é correto afirmar que:

RESPOSTA: B

a) métodos abstratos são aqueles que não devem ser redefinidos em classes derivadas, devem ser herdados tal como foram definidos.
b) métodos estáticos são aqueles que, ao serem executados, não acessam atributos de instância da classe.
c) métodos finais, também conhecidos como finalizadores ou destrutores, são chamados na destruição de uma instância.
d) métodos construtores são métodos chamados sobre um objeto quando ele é criado. Em Java, os construtores têm o mesmo nome da classe da qual são membros e o tipo retornado por eles é especificado na sua definição.
e) métodos de classe são aqueles que executam operações que afetam objetos individuais da classe.

5.) (1 ponto) A programação orientada a objetos é baseada em diversos conceitos, tais como encapsulamento, herança, polimorfismo e abstração. Com relação a esses conceitos, assinale com V para verdadeiro e F para Falso as alternativas abaixo:

a) (**F**) o conceito de encapsulamento é alcançado por meio da definição da visibilidade pública aos atributos e métodos.

b) (**V**) herança é um mecanismo que permite que uma classe herde todo o comportamento e os atributos de outra classe.

c) (**F**) interface pode ser considerada como a forma com que um objeto se apresenta para outros, no que diz respeito aos seus atributos e métodos. Em Java, uma mesma classe não pode implementar mais de uma interface.

d) (**V**) polimorfismo é o uso de um mesmo nome para identificar diferentes implementações dos métodos. Seu uso é comum na definição de construtores, em que os mesmos podem ser implementados em diferentes versões para as diferentes formas de se instanciar a classe.

e) (**F**) para uma classe ser considerada abstrata, todos os seus métodos devem ser abstratos. Em Java, para se definir uma classe abstrata deve-se utilizar a palavra chave "*abstract*" no início de sua declaração

6.) (1 ponto) Para representar quais técnicas da orientação a objeto é utilizada as palavras reservadas *implements* e *extends*?

RESPOSTA:
As palavras reservadas *implements* e *extends* são utilizadas nas técnicas de Polimorfismo e respectivamente Herança.

7.) (1 ponto) Exemplifique utilizando código Java como é construído uma interface e como ela deve ser utilizada em uma classe concreta?

RESPOSTA:

//Interface

```
package br.edu.unoesc.ExemploInterface;

public interface IAnimal {
	public String fazerBarulho();
	public String comer();
	public String dormir();
}
```

//Implementação

```
package br.edu.unoesc.ExemploInterface;

public class Animal implements IAnimal {

	private String som;

	public void setSom(String som) {
		this.som  = som;
	}

	@Override
	public String fazerBarulho() {
		return som;
	}

	@Override
	public String comer() {
		return "Nhack... nhack..";
	}

	@Override
	public String dormir() {
		return	"ZZzzzz...";
	}
}
```

//Utilização

```
package br.edu.unoesc.ExemploInterface;

public class APombo {

	public void acoesAnimal(Animal animal) {
		System.out.println("Comendo... " + animal.comer());
		System.out.println("Dormindo... " + animal.dormir());
		System.out.println("Fazendo Barulho... " +
animal.fazerBarulho());
	}

	public static void main(String[] args) {
		Animal pombo = new Animal();
		pombo.setSom("pruuuuu... pruuuu.. pruuu....");

		APombo acoespombo = new APombo();
		acoespombo.acoesAnimal(pombo);
	}
}
```

//Saída

```
Comendo... Nhack... nhack..
Dormindo... ZZzzzz...
Fazendo Barulho... pruuuuu... pruuuu.. pruuu....
```

8.) (3 pontos) Elabore o código Orientada a Objeto para a regras abaixo e utilize Testes unitários para validar as regras:

Criar um sistema para controlar um Carrinho de compra:

1. O carrinho deve aceitar incluir um ou mais produto e sua quantidade;
2. O valor unitário do item é o valor cadastrado no produto;
3. Deverá ser possível ordenar os itens comprados pelo valor em ordem crescente ou decrescente;
4. Deverá ser possível totalizar o valor do carrinho;
5. Se o usuário comprar produtos do tipo Bebida deverá ter um desconto automático de 5% e se for Vestuário 25%.

RESPOSTA: https://github.com/marcelogarbin/java-avaliacao-poo