PUCPR – Pontifícia Universidade Católica do Paraná
Programação Orientada a Objetos

Engenharia de Computação
Avaliação Formativa 2 – 17.jun.2024

Estudante:

_____

1 – Quanto ao polimorfismo, é VERDADEIRO:

a) O polimorfismo exige o uso de classes abstratas.
b) O polimorfismo pode gerar um stack overflow, por isso deve ser empregado com cuidado.
c) O polimorfismo indica o princípio de que o comportamento pode variar, dependendo do tipo real de um objeto.
d) O polimorfismo depende exclusivamente do fato de podermos sobrecarregar métodos.
e) O polimorfismo não pode ser obtido usando interfaces.

2 – Quanto a Interfaces, é VERDADEIRO:

a) Uma interface declara um conjunto de métodos e suas assinaturas, tal qual uma classe, pode conter implementações concretas que podem ser redefinidas.
b) Uma interface declara um conjunto de métodos e deve ter pelo menos um construtor explícito.
c) Uma interface declara um conjunto de métodos e deve ter pelo menos um construtor concreto, implementado para ser derivado de Object.
d) Uma interface declara um conjunto de métodos e suas assinaturas, ao contrário de uma classe, não fornece nenhuma implementação
e) Uma interface é igual a uma classe abstrata, pode ter construtor mas não ser instanciada.

3 - Considerando o seguinte início de uma classe:

Interface

```
public class CF extends CP implements CM {
```

Superclasse

É VERDADEIRO que:

a) CP é uma interface e CM uma classe abstrata.
b) CP é uma classe e CM uma interface.
c) CP é uma classe abstrata e CM uma classe concreta.
d) CP é uma interface e CM uma interface também.
e) O código não compila, pois não é possível usar implements e extends ao mesmo tempo.

4 - Analise as seguintes afirmativas.

Polimorfismo

Herança

I. Encapsulamento é a capacidade de uma operação atuar de modos diversos em classes diferentes.
II. Polimorfismo é o compartilhamento de atributos e métodos entre classes com base em um relacionamento hierárquico.

Encapsulamento

III. Herança consiste no processo de ocultação dos detalhes internos de implementação de um objeto.
IV. Em Java, todos os métodos numa classe abstrata devem ser declarados como abstratos.
V. Overriding é a redefinição de um método herdado. Os métodos apresentam assinaturas iguais.

A partir da análise, pode-se concluir que:

a) apenas as afirmativas III e IV estão corretas.
b) apenas a afirmativa IV está correta.
c) apenas as afirmativas III e V estão corretas.
d) todas as afirmativas são falsas.
e) apenas a afirmativa V está correta.

5 – Um ArrayList declarado da seguinte forma:

```
ArrayList<Animal> zoo = new ArrayList<Animal>();
```

a) Pode conter objetos da classe Object.
b) Pode conter unicamente objetos da classe Animal.
c) Pode conter qualquer objeto do framework Collections.
d) Pode armazenar unicamente 1 animal, pois foi criado sem tamanho definido.
e) Pode conter objetos da classe Animal e de todas as suas subclasses.

6 – Os modificadores de acesso em Java são:

a) Private, public e protected.
b) Private, public, static e final.
c) Public, final, static e void.
d) Private, public, protected e sem modificador (package private ou acesso de pacote).
e) Private, public, protected e void.

7 – É VERDADEIRO que:

a) Em Java podemos ter herança múltipla.
b) Em Java podemos ter classes anônimas
c) Em Java o código é compilado para um arquivo executável e só pode ser executado no mesmo Sistema Operacional da compilação.
d) A orientação a objetos do Java exige a criação de destrutores explícitos.
e) Java é uma linguagem pouco portável, associada apenas a servidores Windows.

8 – Quanto a tratamento de exceções em JAVA, é FALSO que:

a) Existem dois tipos de exceções: exceções verificadas e exceções não-verificadas.
b) As exceções não-verificadas estendem a classe RuntimeException ou Error.
c) As exceções verificadas se devem a circunstâncias externas que o programador não pode evitar.
d) As exceções verificadas, se ocorrerem, são culpa do programador, por isso devo usar um if, não try-catch.
e) O compilador verifica se seu programa gerencia as exceções verificadas.

Não verificadas.

9 - Analise as seguintes afirmativas.

I. É necessário fazer coerção (typecasting) para converter entre um tipo interface e um tipo classe.
II. A vinculação inicial dos métodos ocorre se o compilador selecionar um método entre os vários possíveis candidatos.
III. A vinculação tardia ocorre se a seleção do método acontecer quando o programa é executado.
IV. Pode-se converter de um tipo classe em um tipo interface desde que a classe implemente a interface.
V. Diferentemente de uma classe, um tipo interface não fornece nenhuma implementação.

A partir da análise, pode-se concluir que:

a) apenas as afirmativas II e III estão corretas.
b) apenas as afirmativas I, II e IV está correta.
c) apenas as afirmativas I, II, III e V estão corretas.
d) todas as afirmativas são corretas.
e) apenas a afirmativa V está correta.

10 - É FALSO que:

a) Uma subclasse sempre tem acesso aos campos privados de sua superclasse.
b) Cada classe estende a classe Object, direta ou indiretamente.
c) Herdar a herança de uma classe é diferente de implementar uma interface: a subclasse herda o comportamento e o estado da superclasse.
d) Para chamar o construtor da superclasse, utiliza-se a palavra-chave super() na primeira instrução do construtor da subclasse.
e) A herança é um mecanismo para estender classes existentes adicionando métodos e campos.