**Polimorfismo**

1. O que é polimorfismo? Descreva com suas palavras em uma frase.

Métodos com a mesma assinatura da superclasse com implementações distintas em subclasses.

1. Explique porque o polimorfismo pode ser uma estratégia útil.

Porque qualquer modificação futura que adicionam novas subclasses não afetarão o código ou seu funcionamento.

1. Em Java, declaramos uma variável usando a sintaxe [Tipo] [variável] = [valor]. Considerando a representação de herança onde Animal é a superclasse e Cachorro a subclasse, quais são TODAS as formas possíveis de declarar uma variável para a nova instância new Cachorro()?

Cachorro cachorro = new Cachorro()

Animal cachorro = new Cachorro()

1. É possível aplicar o princípio do polimorfismo tanto para interfaces quanto para classes. a) V b) F

V

1. Quando definimos uma relação de herança entre duas classes, a classe-filha possui disponível os métodos herdados de sua classe-pai mais os métodos definidos nela mesmo. a) V b) F

V

1. Quando usamos polimorfismo, os métodos específicos da classe-filha ficam inacessíveis, pois estamos representando aquele objeto através da classe-pai. a) V b) F

F

1. Considere a interface Animal com os métodos alimentar e respirar. Considere também a classe Cachorro que implementa a interface Animal, e define seus próprios métodos brincar e comunicar. Quais métodos estarão disponíveis ao declarar a variável Cachorro cachorro = new Cachorro?

Alimentar, respirar, brincar e comunicar

1. Criamos uma classe Cachorro com os seus métodos próprios brincar e comunicar. Além disso, Cachorro implementa a interface Animal que possui definição para o método alimentar. Se aplicarmos o princípio do polimorfismo declarando uma variável do tipo Animal cachorro = new Cachorro(), quais métodos estarão disponíveis na variável cachorro?

Alimentar, brincar e comunicar

1. Ao criar uma classe chamada Aluno e invocar que uma variável Aluno aluno = new Aluno(); para exibir em console através de System.out.println(aluno);, o resultado é semelhante a Aluno@6a6824be. Por quê?

Porque ele herda automaticamente a escrita da classe Object. Para que o resultado seja diferente é necessário usar o método toString() que sobrescreve o modelo da classe Object.

1. Ao declarar uma variável do tipo Aluno aluno = new Aluno();, e exibi-la em console através de System.out.println(aluno); o resultado foi Aluno@6a6824be. Considerando que a classe Aluno possui os atributos nome e matrícula, como poderíamos definir um método que exiba corretamente essas informações em console?

Texto

Descrição gerada automaticamente

**Enviar respostas via e-mail:**[**rodolfo.lima.ferreira11@gmail.com**](mailto:rodolfo.lima.ferreira11@gmail.com)