

Algoritmos e Estruturas de Dados I
Aula06

Polimorfismo

Prof. MSc. Adalto Selau Sparremerger

 assparremerger@senacrs.com.br

  @adaltoss
 

 /assparremerger

Polimorfismo

- ▷ Poli => muitos
- ▷ Morfo => formas
- ▷ Polimorfismo => Permite que um mesmo nome represente vários componentes diferentes
- ▷ Derivação grega de algo que assume muitas formas

Polimorfismo – Overloading ou Sobrecarga (Java)

- ▷ O termo sobrecarga vem do fato de declararmos vários métodos com o mesmo nome, estamos carregando o aplicativo com o 'mesmo' método.
- ▷ A única diferença entre esses métodos são seus parâmetros e/ou tipo de retorno
- ▷ Por exemplo: vamos declarar, num mesmo aplicativo, dois métodos quadrados: um que recebe e retorna inteiros e outro que recebe e retorna double.

Polimorfismo – Overloading ou Sobrecarga (Java)

- ▷ Quando declaramos estes dois métodos, estamos fazendo uma sobrecarga de métodos, ou method overloading em Java.
- ▷ O que acontece, então, quando fazemos o chamado do método, já que existem duas?
- ▷ Embora os nomes sejam os mesmos, eles atuam de forma diferentes e ocupam espaços diferentes em memória, pois estão lidando com tipos diferentes de variáveis.
- ▷ Quando invocamos o método com um inteiro como um argumento, o Java é inteligente o suficiente para invocar corretamente o método que foi declarado com inteiro como parâmetro.
- ▷ Caso invoquemos o método usando um double como um argumento, o método a ser executado será aquele que foi declarado com o tipo double em seu parâmetro.

Polimorfismo – Overloading ou Sobrecarga (Python)

- ▷ Python não tem suporte a sobrecarga.
- ▷ Em Python, métodos se tornam atributos dos objetos, então não é possível criar métodos com o mesmo nome.
- ▷ É possível utilizar parâmetros com valores default, tornando-os opcionais.

Polimorfismo – Overloading ou Sobrecarga

▷ Em Java

```
public int somar(int x){  
    return x + 0;  
}  
  
public double somar(double x){  
    return x + 0;  
}  
  
public int somar(int x, int y){  
    return x + y  
}  
  
public double somar(double x, double y){  
    return x + y  
}
```

▷ Em Python

```
def somar(self, x = None , y = None):  
    if x == None and y == None:  
        return None  
    elif y == None:  
        return x  
    else:  
        return x + y
```

Polimorfismo – Sobrescrita ou Sobreposição (Java)

- ▷ Sobrescrever métodos é como herdar habilidades dos seus pais, mas dar a sua personalidade para essas habilidades. Meu pai foi quem me ensinou a dirigir. Apesar de ter aprendido com ele, eu não dirijo exatamente como ele. Eu tenho alguns comportamentos diferentes dos dele, principalmente em relação à forma de estacionar. Então, eu e meu pai sabemos como estacionar um carro, mas eu faço isso de uma forma diferente da que ele me ensinou.
- ▷ Redefinir o comportamento herdado de uma classe é chamado de sobrescrita de método. Os métodos sobrescritos também são chamados de métodos polimórficos.
- ▷ Uma classe só pode sobrescrever os métodos da classe mãe aos quais ela tem acesso.

Polimorfismo – Sobreescrita ou Sobreposição (Java)

```
public class Filme {  
  
    public void alugarFilme(int dias){  
        if ((dias > 0) && (dias <= 5)){  
            System.out.println("Aluguel feito.");  
        }else{  
            System.out.println("Não é possível alugar um filme por menos de 0 dias ou mais de 5 dias.");  
        }  
    }  
}
```

```
public class Filme24Horas extends Filme {  
  
    @Override  
    public void alugarFilme(int dias) {  
        if ((dias > 0) && (dias <=1)){  
            System.out.println("Aluguel feito.");  
        }else{  
            System.out.println("Filme 24 horas deve ser alugado por no máximo 1 dia.");  
        }  
    }  
}
```


Polimorfismo – Sobrescrita ou Sobreposição (Python)

```
class Filme:  
    def alugarFilme(self, dias):  
        if dias > 0 and dias <= 5 :  
            print("Aluguel efetuado.")  
        else:  
            print("Não é possível alugar um filme por menos de 0 dias ou mais de 5 dias.")
```

```
class Filme24Horas(Filme):  
    def alugarFilme(self, dias):  
        if dias > 0 and dias <= 1 :  
            print("Aluguel efetuado.")  
        else:  
            print("Filme 24 horas deve ser alugado por no máximo 1 dia.")
```

Polimorfismo – Sobrescrita ou Sobreposição (Python)

```
class Pessoa:

    def __init__(self, nome, endereco):
        self.nome = nome
        self.endereco = endereco

    def imprimir(self):
        print("Nome: " + self.nome)
        print("Endereco: " + self.endereco)
```

```
class Fisica(Pessoa):


    def __init__(self, nomeF, enderecoF, cpfF):
        Pessoa.__init__(self, nomeF, enderecoF)
        self.cpf = cpfF

    def imprimir(self):
        Pessoa.imprimir(self)
        print("CPF: " + self.cpf)
```

Referências

- ▶ BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML**. Rio de Janeiro: Elsevier, 2007.
- ▶ W3Schools: <https://www.w3schools.com/python/default.asp>

Material extra:

- ▶  Vídeo1 : Conceito Polimorfismo (Parte 1)
Conceito Sobrescrita: <https://www.youtube.com/watch?v=9-3-RMEMcg4>
Implementação em Java: <https://www.youtube.com/watch?v=NctjqlfKC0U&t=558s>
- ▶  Vídeo 2: Conceito Polimorfismo (Parte 2)
Conceito Sobrecarga: <https://www.youtube.com/watch?v=hYek1xqWzgs>
Implementação em Java: <https://www.youtube.com/watch?v=b7xGYh3NHZU>

Exercício

▷ Implementar as classes do diagrama a seguir:

- Crie o construtor para cada uma das classes
- O método acelerar da classe Veículo deve somar o valor passado por parâmetro da velocidadeAtual do veículo
- O método frear da classe Veículo deve subtrair o valor passado por parâmetro da velocidadeAtual do veículo.
- O método imprimirInformacoes de cada uma das classes deve exibir na tela o conteúdo de cada um dos atributos da classe.

