Simplicidade com Orientação a Objetos

Módulo VIII

Leonardo Flores

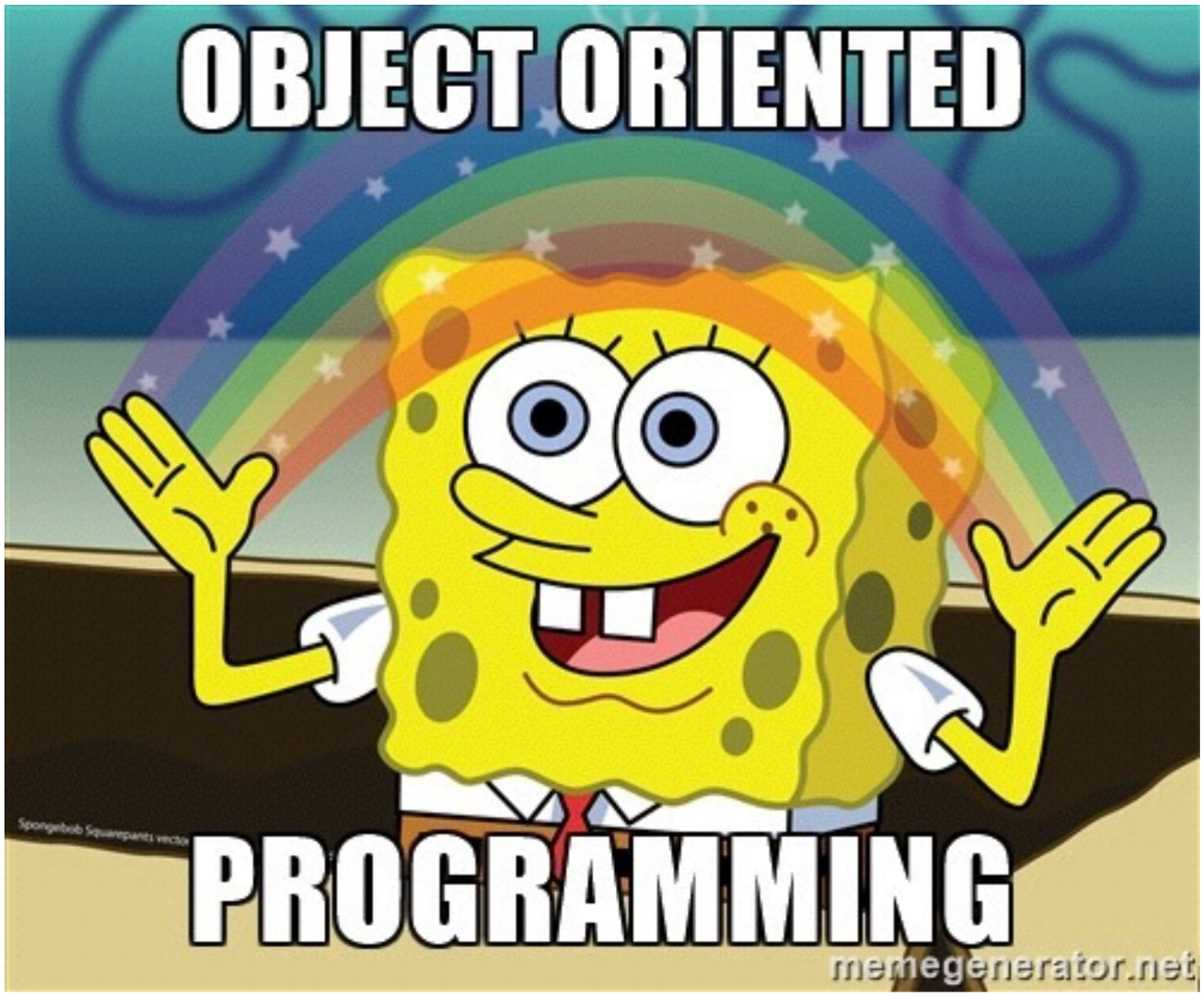
leonardocouy@hotmail.com

Introdução

- Paradigma que facilita a manutenção, torna o código reutilizável, extensível e compatível.

Leonardo Flores leonardocouy@hotmail.com

Introdução



Leonardo Flores leonardocouy@hotmail.com

Introdução

- Entidades são chamadas de objetos. Os objetos conhecem muito bem a si mesmos e respondem as mensagens de acordo com seus atributos (suas caracteristicas) e com seus próprios métodos (funções)

Leonardo Flores **leonardocouy@hotmail.com**

Conceitos

- **Classe:** Representa um tipo especifico de objeto, imagine que seja um FABRICANTE DE OBJETOS, PRODUZ INSTANCIAS. EX: Categoria, tipo de objeto, algo genérico
  - Exemplo: Profissão, Cão, Pessoa
  - Podemos imaginar genericamente o que cada classe dessa poderia ter (Pessoa: tem nome, idade, identidade)
  - É uma estrutura que visa abstrair e imitar o comportamento de coisas reais.

Conceitos

- **Objeto:** tipo abstrato que contém dados mais os procedimentos que manipulam esses dados ou é uma unidade de software que encapsula algoritmos e os dados sobre o qual os algoritmos atuam

Conceitos

- **Atributos:** São objetos de uma classe. Cada objeto utilizado em uma aplicação pertencente a uma classe é uma instância dessa classe (Objeto fabricado pela Classe é uma instância)

Conceitos

- **Métodos:** Procedimentos ou funções residentes nos objetos que determinam como eles irão atuar ao receber as mensagens.

Conceitos

- **Instância:** São objetos de uma classe. Cada objeto utilizado em uma aplicação pertencente a uma classe é uma instância dessa classe (Objeto fabricado pela Classe é uma instância)

Conceitos

- **Herança:** O conceito de uma classe herdar atributos e métodos de outra classe

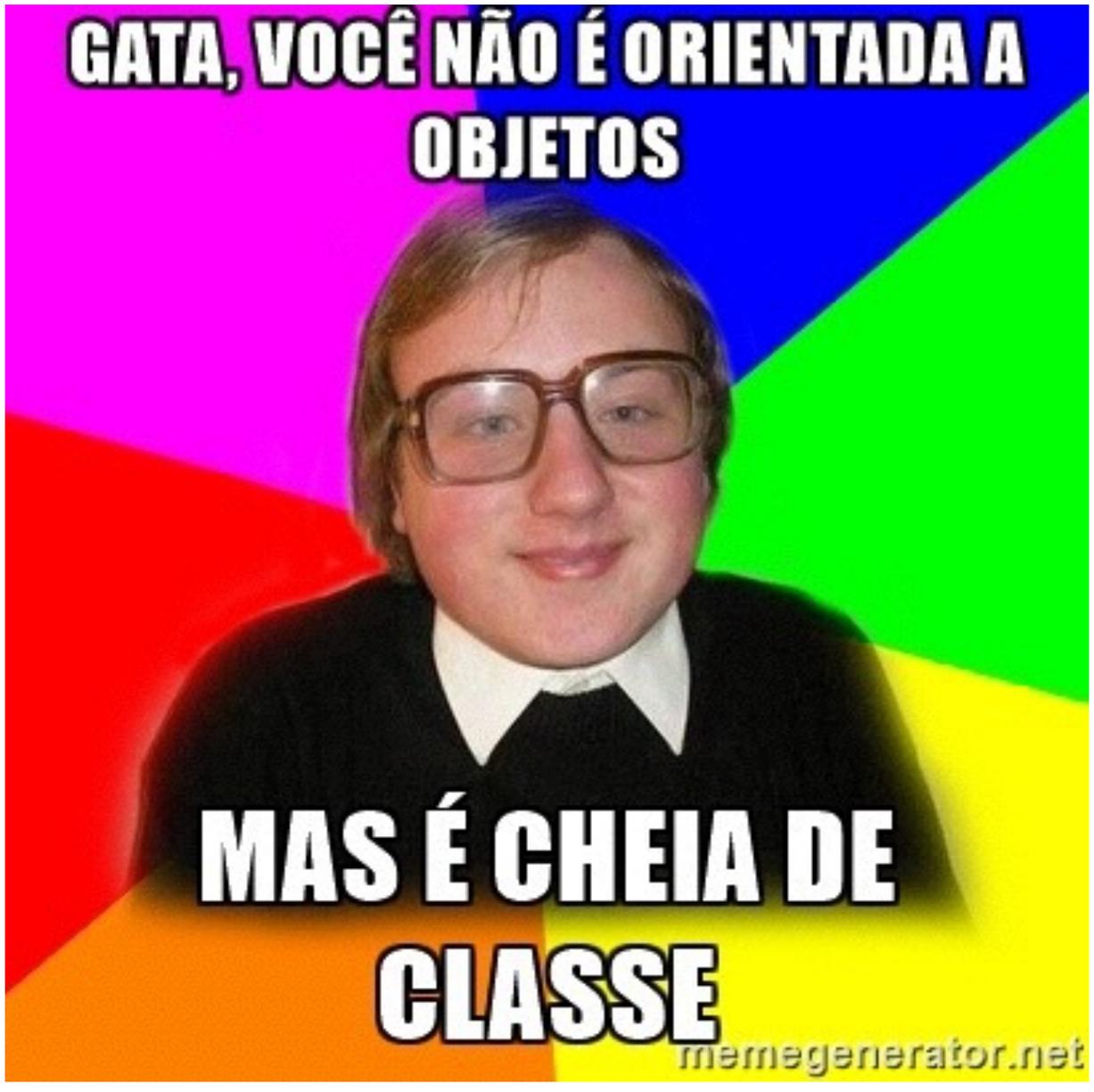
Conceitos

- **Encapsulamento:** O conceito de que uma Classe pode ter atributos/métodos inacessíveis por outras Classes. (os famosos getters and setters em Java)

Conceitos

- **Polimorfismo:** O conceito que objetos de uma mesma hierarquia são tratados como se fossem do mesmo "tipo".

JOKING



Leonardo Flores

leonardocouy@hotmail.com

#Partiu Dominar POO

Leonardo Flores leonardocouy@hotmail.com

```python
class Conta:

    def __init__(self, nome, senha, saldo):
        self.nome = nome
        self.senha = senha
        self.saldo = saldo

    def depositar(self, reais):
        if reais < 0:
            print('Você deve depositar um valor válido')
        else:
            self.saldo = self.saldo + reais

    def sacar(self, reais):
        if self.saldo < reais:
            print("Saldo insuficiente")
        else:
            self.saldo = self.saldo - reais

    def mostrar_saldo(self):
        print(self.saldo)
```

Nome da Classe

Inicializador (construtor)

Se refere a instância/objeto (a si mesmo)

Método (função)

Leonardo Flores

leonardocouy@hotmail.com

```
class Animal:

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

    def falar(self):
        raise NotImplementedError("Sub-classes devem implementar este mâˆšétodo")


class Gato(Animal):
    def falar(self):
        return "{}: Miau, miau!".format(self.nome)


class Cachorro(Animal):
    def falar(self):
        return "{}: Au, au!".format(self.nome)


class Vaca(Animal):
    def falar(self):
        return "{}: Muuuuuuuu...".format(self.nome)


vaca = Vaca('Valentina')
print(vaca.falar())
```

Nome da Classe

Inicializador (construtor)

Método (função)

Herança (Animal é PAI)

Sobreposição do método e polimorfismo (Animal é PAI)

Leonardo Flores

leonardocouy@hotmail.com

Melhor forma de aprender POO

Leonardo Flores leonardocouy@hotmail.com

Referência

- http://pt.slideshare.net/asergionogueira/python-orientao-a-objeto

- http://slides.com/luanfonceca/python-orientado-a-objetos/fullscreen#/8/2