




Implementando Classes em Python

Trabalhando com mais de uma classe

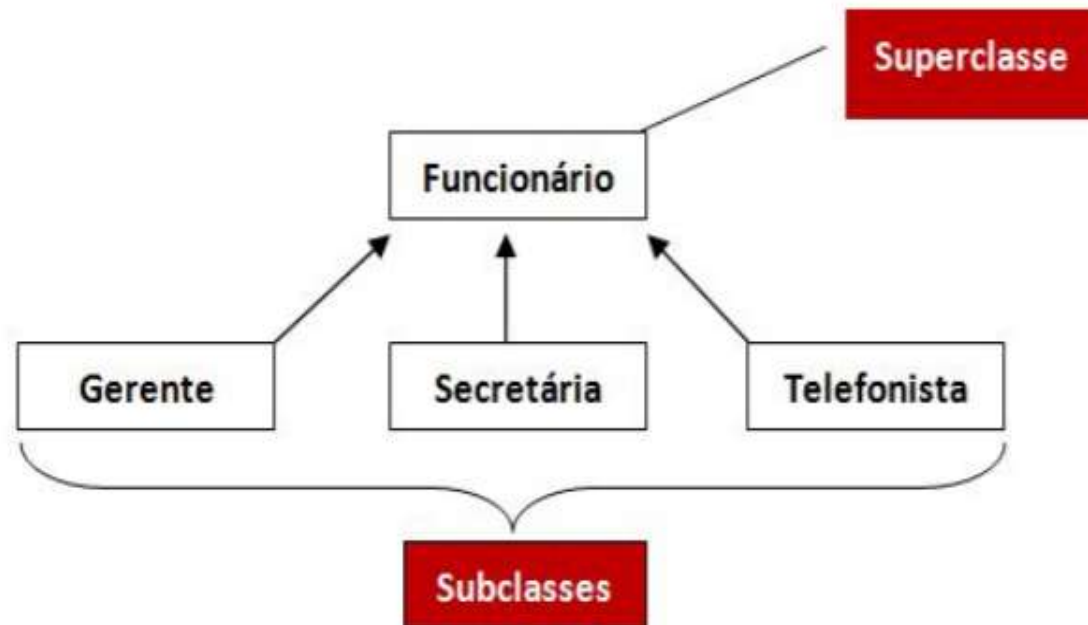
Herança



Herança

- Permite criar uma nova classe a partir de uma já existente.
- O nome herança (ou subclasses), provém do fato de que a subclasse (a classe recém-criada) contém atributos e métodos da classe primária (da qual deriva).
- A principal vantagem da herança é a capacidade para definir novos atributos e métodos para a subclasse, que se somam aos atributos e métodos herdados.
- Esta particularidade permite criar uma estrutura hierárquica de classes cada vez mais especializada. A grande vantagem disso é não ter que partir do zero para especializar uma classe existente.

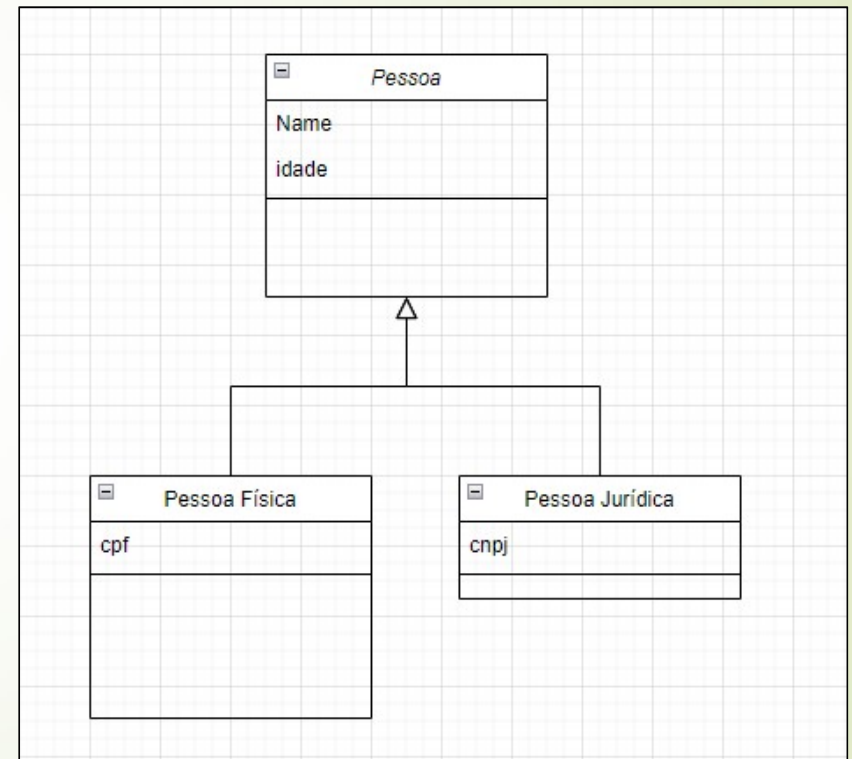
Exemplo 1:





Exemplo 2:

(código fonte disponível no moodle)

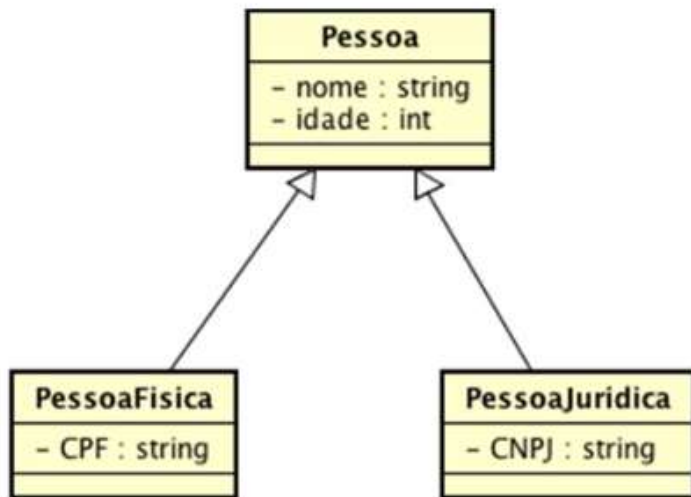
- Implementar um algoritmo para efetuar o cadastro de pessoas.
- 3 propriedades são importantes neste cadastro: nome, idade, identificação (id)
- considerando que uma pessoa pode ser:
 - Pessoa Física (id = cpf)
 - ou
 - Pessoa Jurídica (id = cnpj)






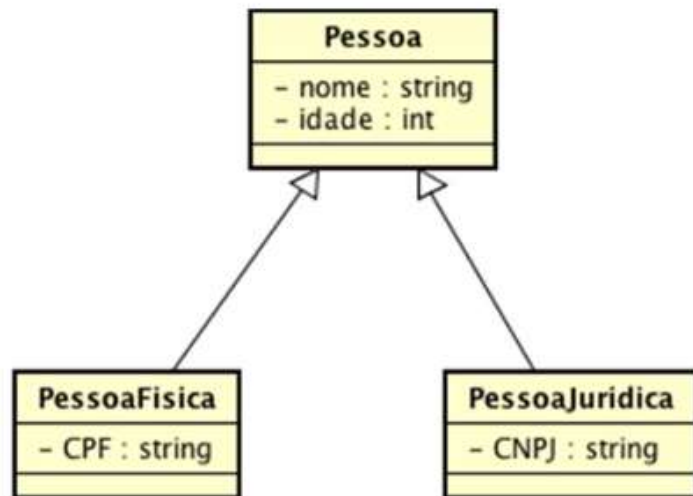
| Pessoa |
|-----------------|
| - nome : string |
| - idade : int |

```
01 class Pessoa:
02     def __init__(self, nome, idade):
03         self.nome = nome
04         self.idade = idade
05
06     def setNome(self, nome):
07         self.nome = nome
08
09     def setIdade(self, idade):
10         self.idade = idade
11
12     def getNome(self):
13         return self.nome
14
15     def getIdade(self):
16         return self.idade
```



```
01 from pessoa import Pessoa
02
03 class PessoaFisica(Pessoa):
04     def __init__(self, cpf, nome, idade):
05         super().__init__(nome, idade)
06         self.CPF = cpf
07
08     def setCPF(self, cpf):
09         self.CPF = cpf
10
11     def getCPF(self):
12         return self.CPF
13
14
15
```

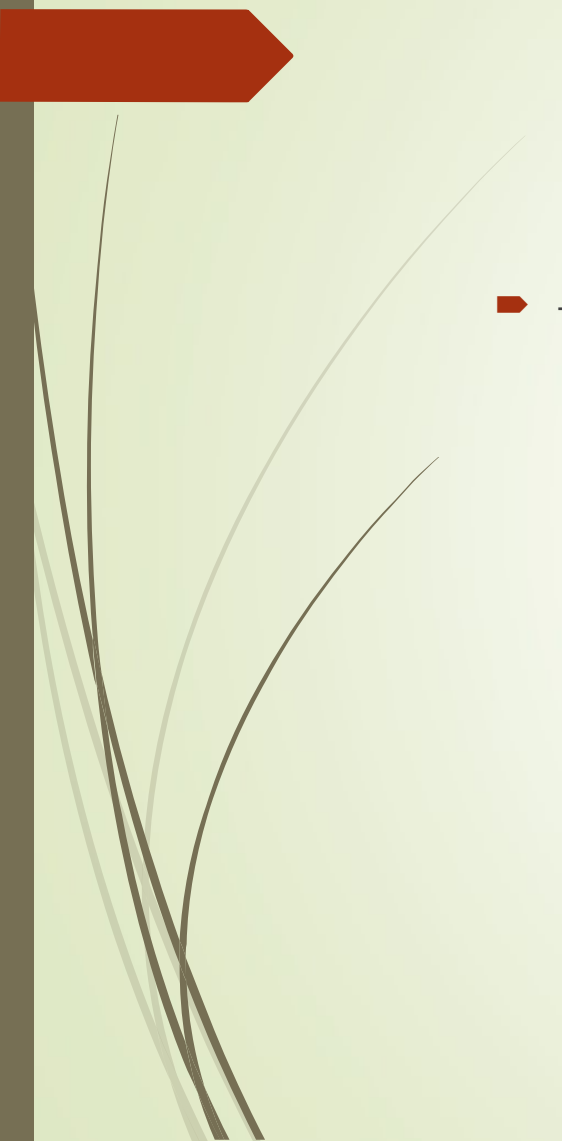




```
1  from pessoa import Pessoa
2
3  class PessoaJuridica(Pessoa):
4      def __init__(self, CNPJ, nome, idade):
5          super().__init__(nome, idade)
6          self.CNPJ = CNPJ
7
8      def getCNPJ(self):
9          return self.CNPJ
10
11     def setCNPJ(self, CNPJ):
12         self.CNPJ = CNPJ
```

Exercício:

- Considere que você foi contratado para desenvolver um sistema de cadastro de clientes de um PetShop. Este PetShop trabalha somente com cães.
- As informações gerais sobre os clientes são (raça, peso, idade, nome do animal, nome do dono, mensalidade).
- Alguns cães possuem cadastro VIP. Para estes o cálculo de valores são diferenciados.
 - Informações específicas sobre clientes VIP:
 - restrição alimentar (boolean),
 - vantagens: 20% desconto pacote de banho e tosa
- Requisitos do sistema:
 - a) Cadastrar cliente (cão);
 - b) Imprimir as informações do cliente cadastrado;
 - c) Criar um método que verifica (retorna) se determinado cão possui restrição alimentar;
 - d) Criar um método que verifique a mensalidade do cachorro (calcular a partir do número de banhos por mês (R\$25,00 por banho))
 - d) Criar um método que verifica qual o cachorro mais idoso.

- 
- ▀ - assistir o vídeo: Implementando Herança (disponível no moodle).
 - ▀ código fonte apresentado no vídeo está disponível na pasta Ex_Herança.