Inatel
Instituto Nacional de Telecomunicações

AC-309 – Atividades Complementares

(Engenharias Biomédica e de Telecomunicações)

6. POO – Herança e Polimorfismo

Prof. Evandro Luís Brandão Gomes

Herança

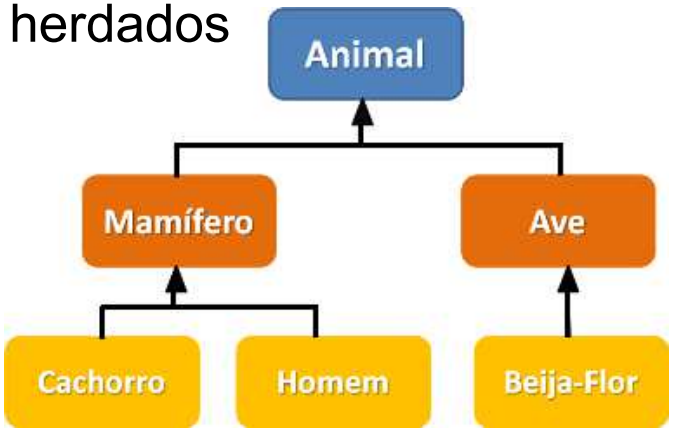
• Permite a hierarquização das classes em um sistema

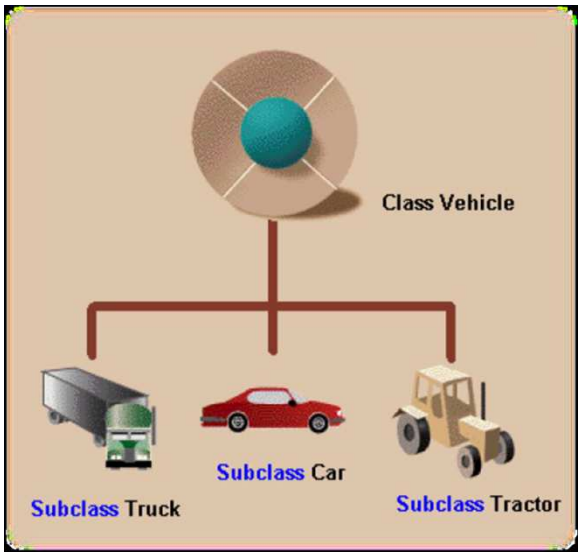
• Uma classe mais especializada (**sub-classe ou classe-derivada**) herda as propriedades (métodos e atributos) de uma classe mais geral (**super-classe ou classe-base**)

• Uma sub-classe pode sobrescrever o comportamento de uma super-classe (**polimorfismo**)

• Novos atributos e métodos podem ser definidos nas sub-classes, além dos herdados

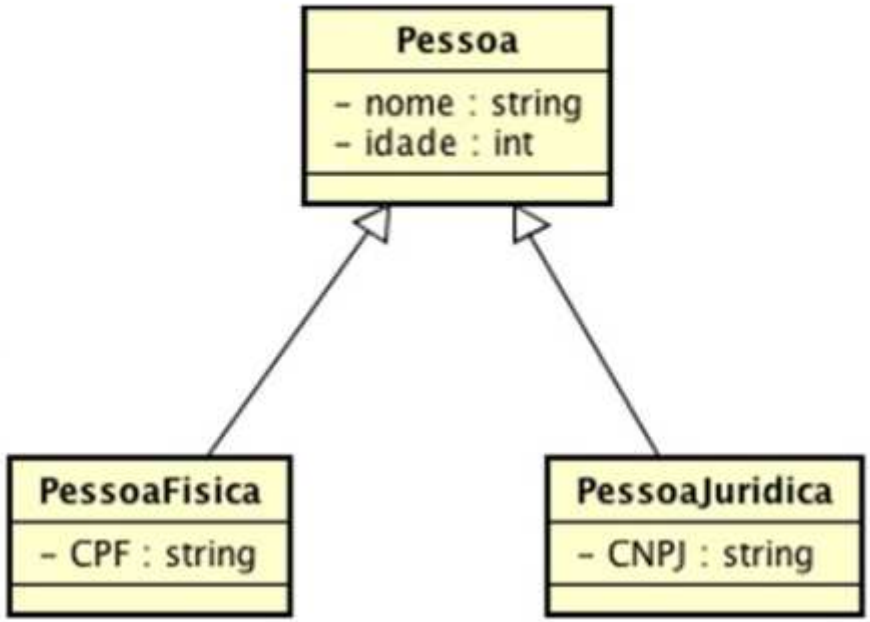
• Promove reuso





Herança

Exemplo

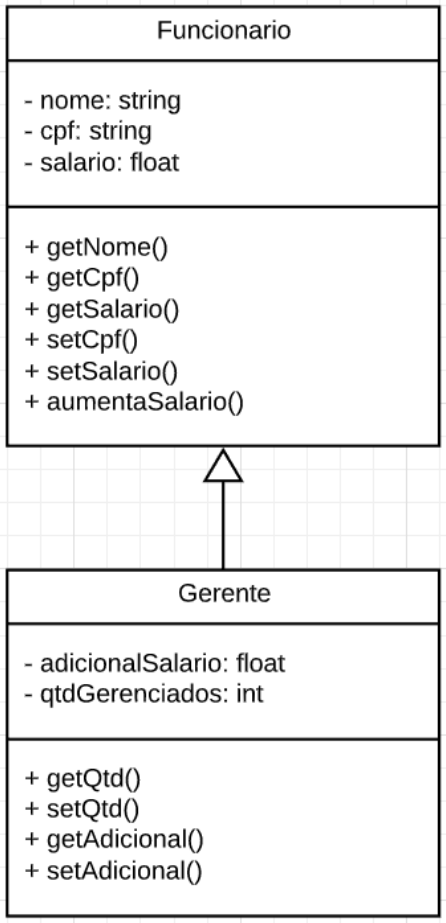



```python
class Pessoa:
    def __init__(self, nome, idade):
        self.nome = nome
        self.idade = idade
    def setNome(self, nome):
        self.nome = nome
    def setIdade(self, idade):
        self.idade = idade
    def getNome(self):
        return self.nome
    def getIdade(self):
        return self.idade

class PessoaFisica(Pessoa):
    def __init__(self, CPF, nome, idade):
        super().__init__(nome, idade)
        self.CPF = CPF
    def getCPF(self):
        return self.CPF
    def setCPF(self, CPF):
        self.CPF = CPF

class PessoaJuridica(Pessoa):
    def __init__(self, CNPJ, nome, idade):
        super().__init__(nome, idade)
        self.CNPJ = CNPJ
    def getCNPJ(self):
        return self.CNPJ
    def setCNPJ(self, CNPJ):
        self.CNPJ = CNPJ
```

O **super()** é usado para fazer referência a superclasse, a classe mãe.

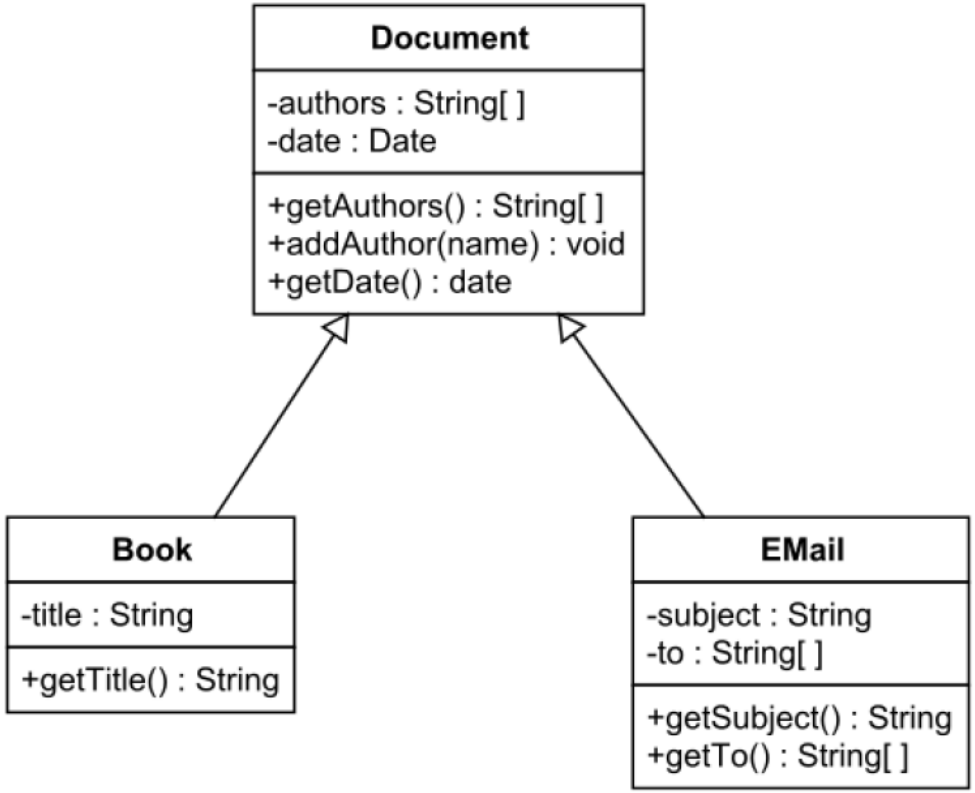
Herança

Exercício: Implemente as classes mostradas no diagrama de classes abaixo e faça o programa de teste.



Herança

Exercício Proposto: Implemente as classes abaixo de acordo com o diagrama de classes e faça o programa de teste.



Polimorfismo

Polimorfismo, em biologia, é um princípio no qual um organismo pode surgir de formas diferentes.

- Indivíduos de uma mesma espécie possuem muitas características similares.

- Contudo, algumas características são peculiares.

Polimorfismo

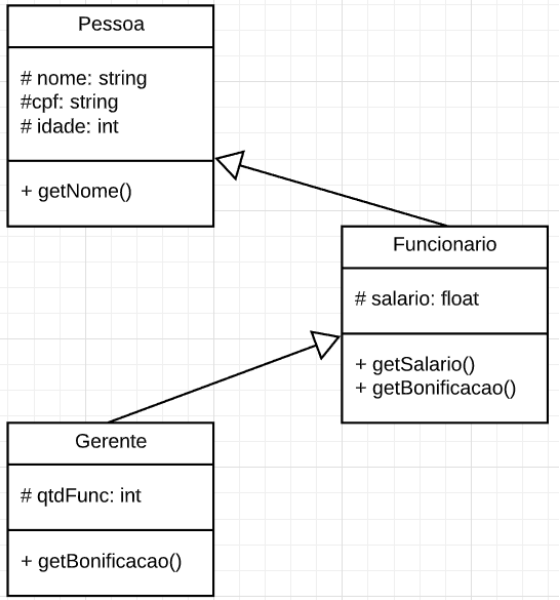
Polimorfismo, em Python, é a capacidade que uma subclasse tem de ter métodos com o mesmo nome de sua superclasse, e o programa saber qual método deve ser invocado, especificamente (da super ou sub).

Ou seja, o objeto tem a capacidade de assumir diferentes formas (polimorfismo).

Em Python, o polimorfismo também é conhecido como Reescrita de Método.

Polimorfismo

Exemplo:




```python
class Pessoa:
    def __init__(self, nome, cpf, idade):
        self._nome = nome
        self._cpf = cpf
        self._idade = idade
    def getNome(self):
        return self._nome

```

```python
class Funcionario(Pessoa):
    def __init__(self, nome, cpf, idade, salario):
        self._salario = salario
        super().__init__(nome, cpf, idade)
    def getSalario(self):
        return self._salario
    def getBonificacao(self, valor):   #valor em %
        return self._salario * valor/100

class Gerente(Funcionario):
    def __init__(self, nome, cpf, idade, salario, qtd):
        self._qtdfunc = qtd
        super().__init__(nome, cpf, idade, salario)
    def getBonificacao(self, valor):   #valor em %
        if self._qtdfunc <= 10:
            bonus = 1000.00
        elif self._qtdfunc <=20:
            bonus = 2000.00
        else:
            bonus = 3000.00
        return self._salario * valor/100 + bonus
```

Polimorfismo

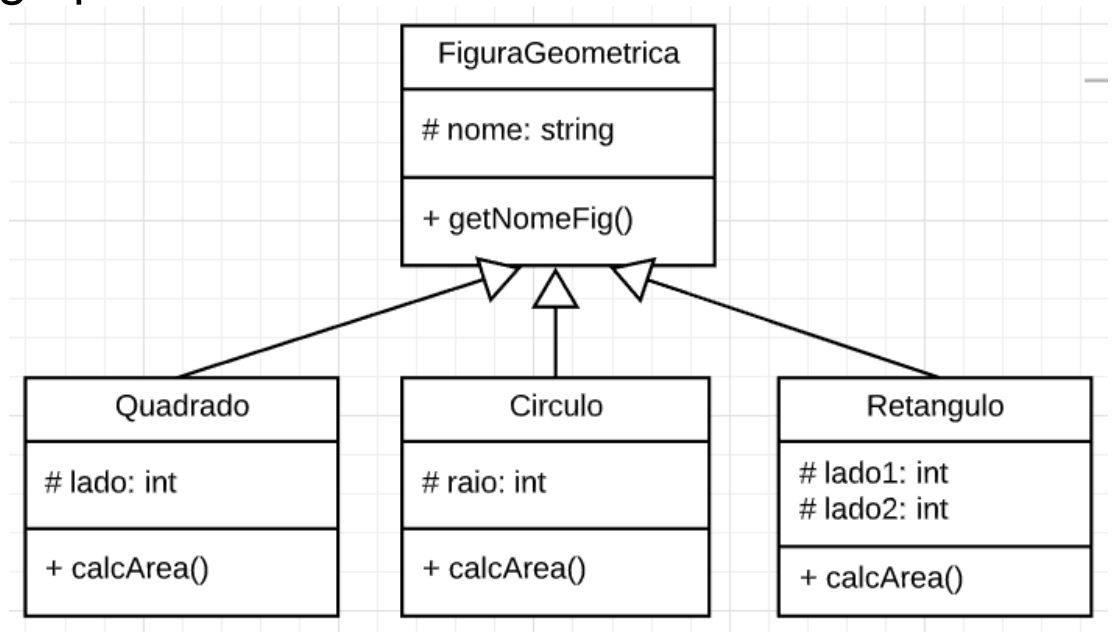
```python
class Pessoa:
    def __init__(self, nome, cpf, idade):
        self._nome = nome
        self._cpf = cpf
        self._idade = idade
    def getNome(self):
        return self._nome

class Funcionario(Pessoa):
    def __init__(self, nome, cpf, idade, salario):
        self._salario = salario
        super().__init__(nome, cpf, idade)
    def getSalario(self):
        return self._salario
    def getBonificacao(self, valor):   #valor em %
        return self._salario * valor/100

class Gerente(Funcionario):
    def __init__(self, nome, cpf, idade, salario, qtd):
        self._qtdfunc = qtd
        super().__init__(nome, cpf, idade, salario)
    def getBonificacao(self, valor):   #valor em %
        if self._qtdfunc <= 10:
            bonus = 1000.00
        elif self._qtdfunc <=20:
            bonus = 2000.00
        else:
            bonus = 3000.00
        return self._salario * valor/100 + bonus
```

```python
p1 = Pessoa("João da Silva", "123.456.789-10", 34)
p2 = Funcionario("José da Silva","111.222.333-44",28,1000.00)
p3 = Gerente("Big Boss","999-888-777-66",45,5000.00,50)
print("Func: %s    Salario=%0.2f    Bonificação=%0.2f" %(p2.getNome(),
                                    p2.getSalario(), p2.getBonificacao(10)))
print("Func: %s    Salario=%0.2f    Bonificação=%0.2f" %(p3.getNome(),
                                    p3.getSalario(), p3.getBonificacao(10)))
```

```
polimorfismo
C:\Users\Evandro\AppData\Local\Programs\Python\Python38\python.exe
Func: José da Silva    Salario=1000.00    Bonificação=100.00
Func: Big Boss    Salario=5000.00    Bonificação=3500.00

Process finished with exit code 0
```

Exercícios:

1) Implementar as classes de acordo com o diagrama de classe a seguir e fazer o código para testar.



2) Implemente novas sub-classes com outras figuras geométricas.