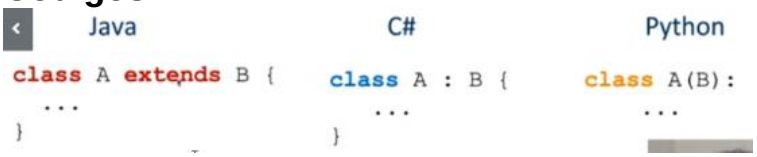
As relações: Herança, Associação e Interface

Herança

“É o relacionamento entre classes em que uma classe chamada de subclasse (classe filha, classe derivada) é uma extensão, um subtipo, de outra classe chamada de superclasse (classe pai, classe mãe, classe base). Devido a isto, a subclasse consegue reaproveitar os atributos e métodos dela. Além dos que venham a ser herdados, a subclasse pode definir seus próprios membros. ”

A herança não é para reuso e sim para criar subtipos, mas específicos e especializados já existentes.

Códigos



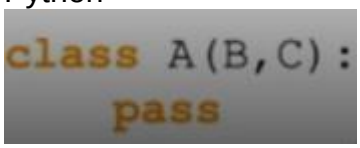
Tipos de Herança

Simples
A classe filha tem só uma classe mãe.

Múltipla
A classe filha tem uma ou mais classes mães.

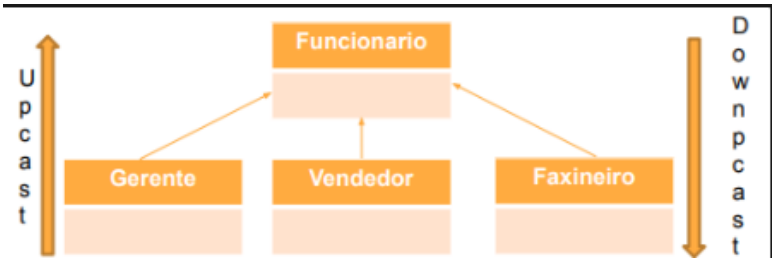
Linguagens:

- Java ⊘
- C# ⊘
    - Ele não tem, pois pode ocorre o:
        - Conflito de Diamante;
        - Conflito de Nomes.
- Python



- C++

Upcast e Downcast



- **Upcast**: Subir na hierarquia.
- **Downcast:** Descer na hierarquia.

Código Upcast
Transformo B em A.

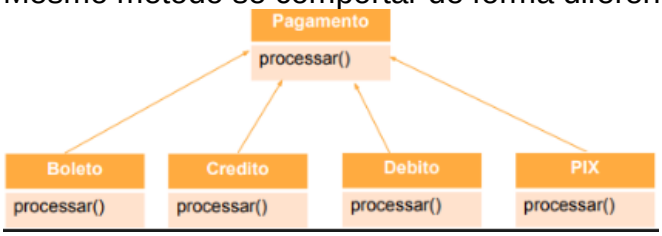
Java
```
A a = new B();
```
C#
```
A a = new B();
```
Python

Código Downcast
Transformo A em B e armazeno em b.

Java
```
B b = (B)new A();
```
C#
```
B b = (B)new A();
```
Python

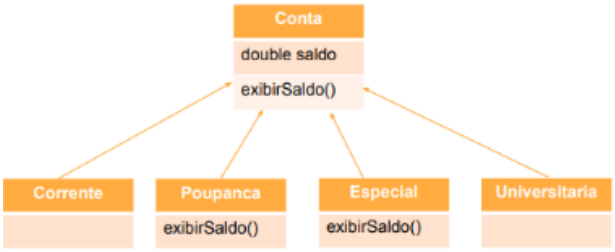
Polimorfismo
"*A mesma ação, se comportando diferente. ”*

Mesmo método se comportar de forma diferente.



O pagamento pode ser feito de formas diferentes.

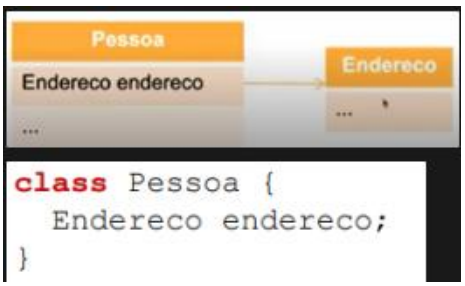
Sobrescrita
“A mesma ação, podendo se comportar diferente. “



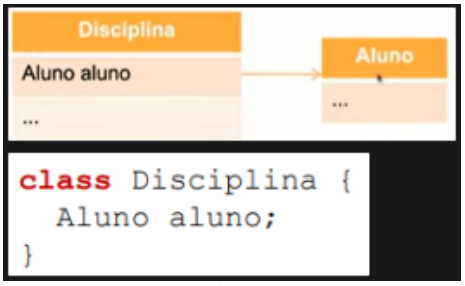
Associação
"Possibilita um relacionamento entre classes/objetos, no qual estes possam pedir ajuda a outras classes/objetos e representar de forma completa o conceito ao qual se destinam. Neste tipo de relacionamento, as classes e os objetos interagem entre si para atingir seus objetivos. ”

Eles interagem entre si, para atingir seu objetivo.

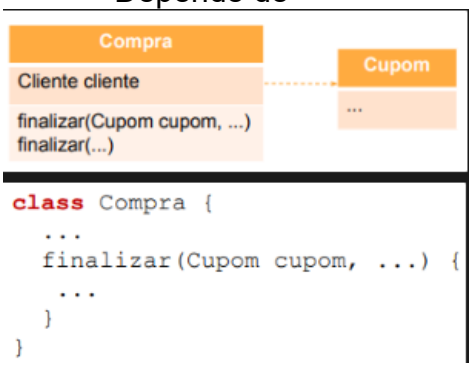
Tipos:
- **Estrutural (Atributos)**
  - **Composição**
    - "Com Parte Todo”
      - Ex: Pessoa e Endereço



- **Agregação**
  - "Sem Parte Todo”
    - Ex: Disciplina e Aluno



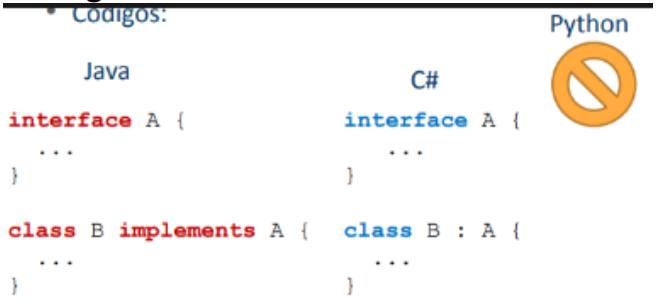
- **Comportamental (Métodos)**
  - **Dependência**
    - "Depende de”



Interface

“*Define um contrato que deve ser seguido pela classe que a implementa. Quando uma classe implementa uma interface, ela se compromete a realizar todos os comportamentos que a interface disponibiliza. ”*

Código



Siga em frente…

Tipos de classe: Abstrata e Concreta
• Métodos abstratos
• Características das associações

• Palavras coringas: super, base e super()
• Relações entre classes e interface: extends e implements