**PBE - Programação Back End**

**Professor Dorival Silva**

Isadora de Abreu Corrêa Nóbrega Lucchesi

**ATIVIDADE – PESQUISA HERANÇA E POLIMORFISMO**

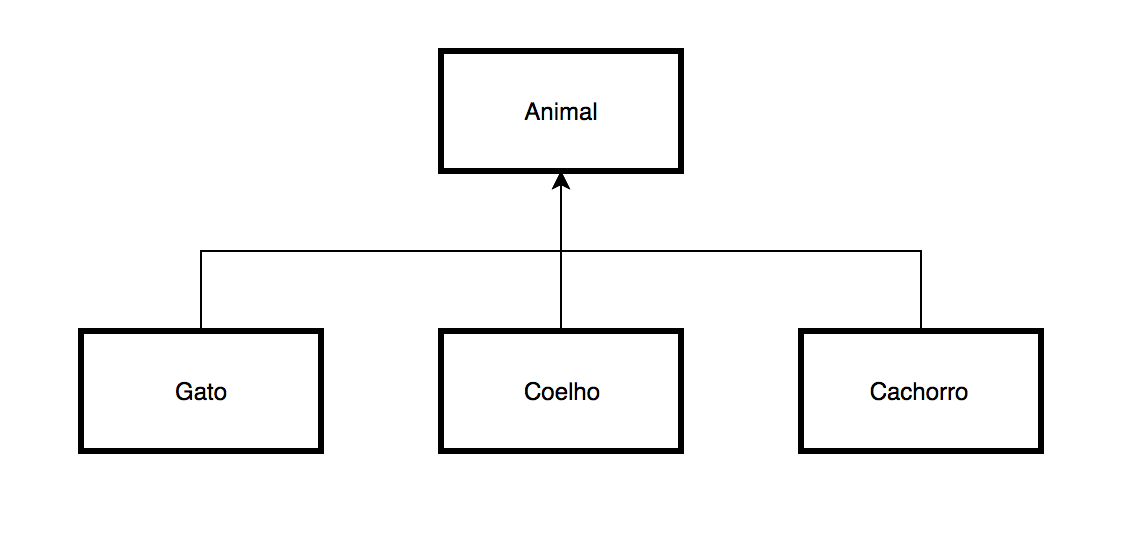
**HERANÇA**

**O QUE É**

Em Python, herança é um conceito fundamental da Programação Orientada a Objetos (POO) que permite que uma classe (classe filha) herde atributos e métodos de outra classe (classe pai).

Ou seja, Classe Pai (Superclasses) é a classe que fornece os atributos para a herança, e Classe Filha (Subclasses) é a que herda atributos e métodos da classe pai. E, por fim, o Python permite que uma classe herde de múltiplas classes (chamada Herança Múltipla), o que pode ser útil para combinar funcionalidades de diferentes classes.

Portanto, quer dizer que a classe filha obtém todos os elementos da classe pai e pode adicionar ou modificar elementos como preferir.

“Animal” = Classe Pai

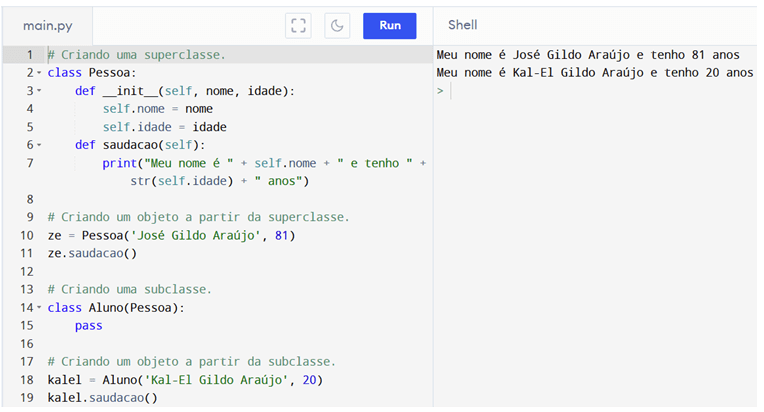
“Gato”, “Coelho” e “Cachorro” - Classe filha

**VANTAGENS**

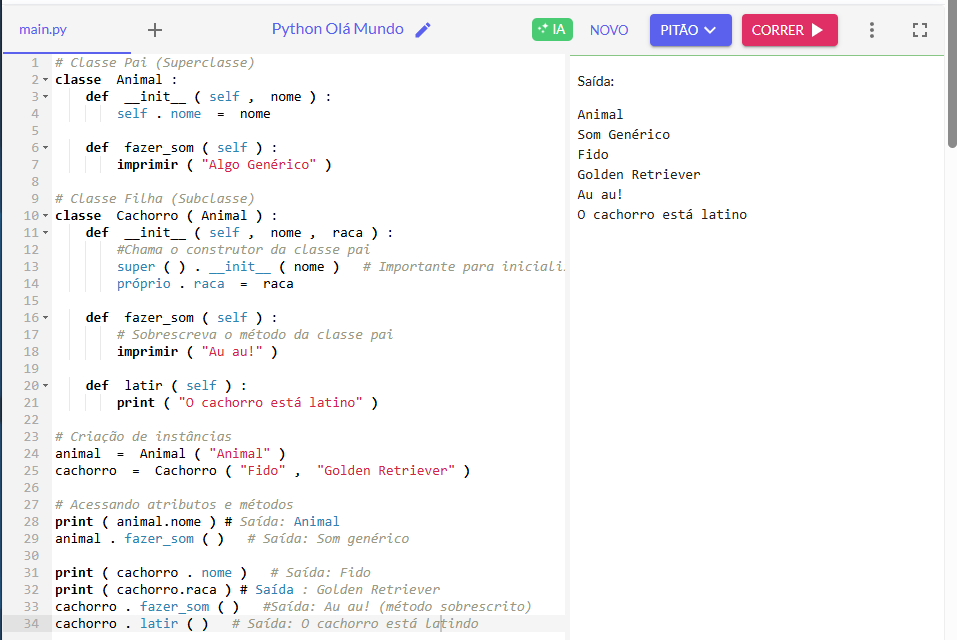
* **Reutilização** - você pode escrever o código uma vez na classe principal e reutilizá-lo nas classes secundárias.
* **Simplicidade** – a herança modela os relacionamentos de forma clara.
* **Escalabilidade** – adiciona novos recursos ou classes secundárias sem afeta o código que existe.
* **Facilidade para manutenção.**
* **Hierarquia de classes** – facilita a organização do código.

**EXEMPLOS PRÁTICOS**

**Exemplo 1:**



**Exemplo 2:**

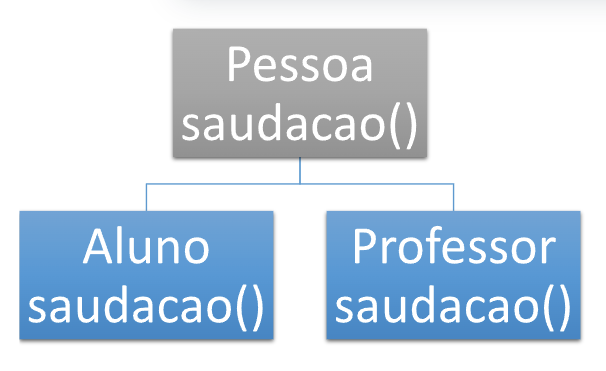


**POLIMORFISMO**

**O QUE É**

Em programação orientada a objetos, polimorfismo significa "muitas formas". Em resumo, é a capacidade de objetos de classes diferentes responderem de forma diferente a um mesmo chamado de método, permitindo que uma única interface seja usada para interagir com objetos de vários tipos. Isso é fundamental para criar código flexível e reutilizável, onde a mesma chamada pode levar a diferentes comportamentos dependendo do tipo do objeto.

Podemos assumir que uma bola de futebol e uma camisa da seleção brasileira são artigos esportivos, mais que o cálculo deles em uma venda é calculado de formas diferentes, por exemplo.



Superclasse = “Pessoa”

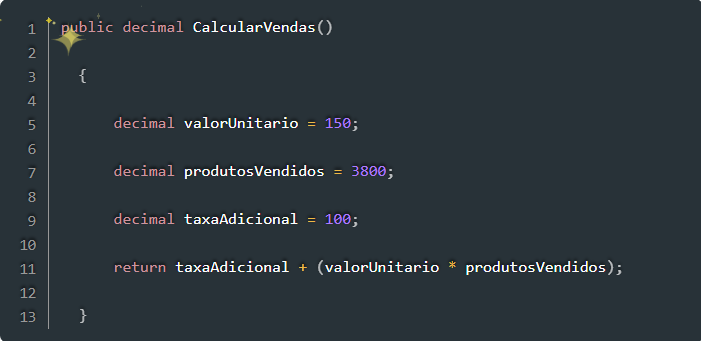
Subclasses = “Aluno” e “Professor”

**VANTAGENS**

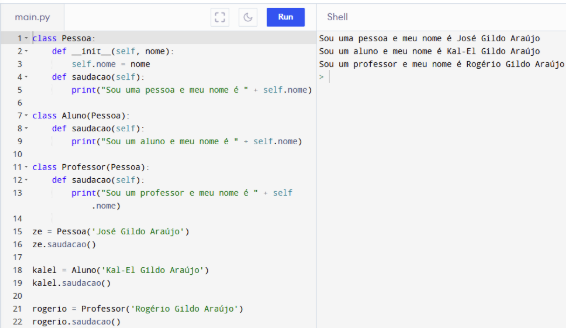
* **Reutilização -** facilita a criação de componentes reutilizáveis.
* **Código mais flexível -** permite adicionar novos tipos de objetos sem alterar o código existente.
* **Facilidade de manutenção -** facilita a expansão do sistema ao logo do tempo.
* **Design mais limpo –** promove uma arquitetura de software mais coesa, com menos acoplamento entre as partes.
* **Código mais legível -** torna o código organizado, mais fácil de entender e manter.

**EXEMPLOS PRÁTICOS**

**Exemplo 1:**



**Exemplo 2:**



**DIFERENÇAS E RELAÇÕES**

* **Relações:**

1. Ambos são princípios da Programação Orientada a Objetos (POO).
2. Trabalham juntos com outros conceitos como encapsulamento e abstração.
3. Ambos promovem reutilização, organização e flexibilidade do código.
4. Ambos envolvem classes e objetos.

* **Diferenças:**

1. Conceito:
   * **Herança:** Uma classe (filha) herda características (métodos e atributos) de outra (pai).
   * **Polimorfismo**: Um mesmo método pode ter comportamentos diferentes dependendo do objeto que o invoca.
2. Objetivo:
   * **Herança:** Reutilizar e organizar código em hierarquias.
   * **Polimorfismo**: Permitir que diferentes classes sejam tratadas de forma uniforme, mas com comportamentos próprios.
3. Foco:

* **Herança:** Estrutura e relação entre classes.
* **Polimorfismo**: Comportamento dos métodos.

**REFERÊNCIAS**

<https://www.datacamp.com/pt/tutorial/python-inheritance>

<https://blog.grancursosonline.com.br/heranca-em-python/>

<https://www.treinaweb.com.br/blog/utilizando-heranca-no-python>

<https://labvital.com.br/glossario/o-que-e-polimorfismo/>

<https://blog.grancursosonline.com.br/polimorfismo-em-python/>