**Polimorfismo** é um dos pilares da Programação Orientada a Objetos (POO) e permite que objetos de diferentes classes possam ser tratados de maneira uniforme, desde que compartilhem métodos ou comportamentos comuns. Em Python, isso significa que várias classes podem ter métodos com o mesmo nome, mas cada uma pode ter sua própria implementação desses métodos.

O termo "polimorfismo" vem do grego e significa "muitas formas". Ele permite que uma mesma interface ou função possa ter comportamentos diferentes dependendo do objeto que a utiliza. Em POO, isso é geralmente implementado por meio de **herança** e **métodos sobrescritos**.

**Exemplificando com o Código Anterior**

Usando o exemplo das classes Pessoa, Cliente e Aluno, vamos aplicar o conceito de polimorfismo.

No código da imagem, cada subclasse (Cliente e Aluno) poderia ter seu próprio comportamento para um método com o mesmo nome, digamos, falar, que foi herdado de Pessoa. Assim, mesmo que chamemos falar em instâncias de Cliente e Aluno, o comportamento pode ser diferente.

Vamos ver isso em prática:



### Explicação

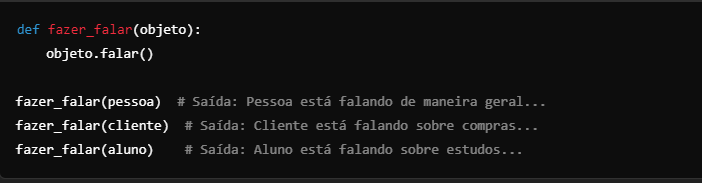
Aqui, cada classe (Pessoa, Cliente e Aluno) tem seu próprio método falar, que exibe uma mensagem diferente. Essa é a **sobrescrita de métodos**, onde subclasses substituem o comportamento do método da superclasse para algo específico a elas.

O polimorfismo permite que possamos chamar falar em qualquer uma das instâncias (Pessoa, Cliente ou Aluno) e obter um comportamento diferente, sem precisar saber a qual classe específica o objeto pertence. Basta invocar falar e o Python chamará automaticamente o método correspondente de acordo com a classe da instância.

### Vantagem do Polimorfismo

A principal vantagem do polimorfismo é que ele permite a criação de código mais genérico e flexível. Podemos escrever funções que funcionam com qualquer objeto que implemente o método esperado, independentemente da classe específica.

Por exemplo, podemos criar uma função que chama o método falar em qualquer objeto Pessoa, Cliente ou Aluno:



Essa função fazer\_falar é genérica e funciona com qualquer objeto que tenha um método falar, demonstrando o poder do polimorfismo em Python.