**Pesquisa POO**

A programação orientada a objetos é um paradigma de programação que, como o nome já diz, organiza seu código em torno de objetos, que representam entidades do mundo real e possuem atributos (características) e comportamentos (métodos), hoje falaremos de duas coisas principais em POO, herança e polimorfismo.

**Conceito**

**Herança:**

Herança é um tipo de relacionamento entre classes, que permite que elas compartilhem atributos e métodos, ela é baseada em um sistema hierárquico onde temos a superclasse ou classe base que é a classe “pai” que dá seus atributos e métodos para classe “filha” chamada de subclasse ou classe derivada.

Podemos dar um exemplo teórico disso, vamos dizer que temos a superclasse cliente, que tem os atributos código e endereço e o método localizar endereço, uma subclasse cliente físico pega os atributos código e endereço da superclasse e adiciona nome e CPF, além do método verificar CPF que ela também cria.

Então temos:

Cliente(superclasse): Atributos: código e endereço

Métodos: localizar endereço

Cliente Físico(subclasse): Atributos: código, endereço, nome e CPF

Métodos: localizar endereço e verificar CPF

**Polimorfismo:**

O polimorfismo é o conceito que permite que diferentes tipos de objeto respondam ao mesmo conjunto de operações de maneiras diferentes, ou seja, podemos definir um método em várias classes diferentes e cada classe pode ter sua implementação e resposta a esse método.

Quando duas ou mais classes derivadas de uma mesma superclasse respondem a um método de formas diferentes é polimorfismo e isso acontece quando as subclasses redefinem esse método dentro delas sobrescrevendo o que havia lá anteriormente.

Trazendo um exemplo mais palpável vamos dizer que temos a superclasse mamífero, e 3 outras classes derivam dela, macaco, baleia e humano, a classe mamífero tem o método locomoverSe, mas cada um dos elementos que herdam ela se locomovem de jeitos diferentes, o macaco pula de galho em galho, a baleia nada e o humano anda, então quando você chamar esse método para cada uma dessas classes você terá 3 respostas diferentes, o que é o polimorfismo.

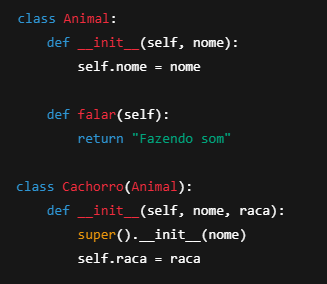
**Diferenças**

A principal diferença entre herança e polimorfismo é que todo polimorfismo é uma herança mas nem toda herança é um polimorfismo, ou seja, para acontecer o polimorfismo o método tem que ser herdado de uma superclasse para duas ou mais subclasses diferentes, mas para ter uma herança não necessariamente eu preciso fazer um polimorfismo, partindo do ponto de vista mais prático o polimorfismo só se aplica a métodos já a herança se aplica a métodos e atributos, mas elas não são excludentes, elas se completam, pois no polimorfismo ao herdar um método eu necessariamente preciso sobrescrever ele para se adaptar a minha subclasse, me retornando algo diferente da resposta do mesmo em outra subclasse, já na herança eu não mudo o que o método faz, se eu chamar ele aqui ou em outro lugar ele vai fazer a mesma coisa.

Então os dois são coisas até parecidas, mas em seu cerne bem diferentes, elas se complementam e funcionam entre si, fazendo um código melhor.

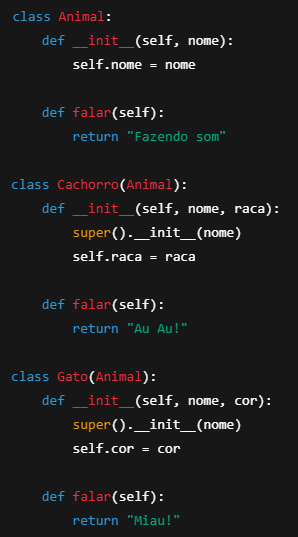
**Exemplo prático**

**Herança:**

****

Neste exemplo temos a classe animal com atributo nome e o método falar, em baixo dela temos uma subclasse cachorro que herda ela, pegando seu atributo nome e também colocando um novo atributo raça.

**Polimorfismo:**

****

Já no polimorfismo podemos ver que as duas classes herdam os atributos e o método de Animal, porém quando escrito o método na classe cachorro ao invés de “Fazendo som” foi sobrescrito para “Au au!”, a mesma coisa aconteceu com a subclasse gato, porém com “Miau!” no lugar, ou seja, se eu chamar o mesmo método para os dois, um vai me devolver “Au au!” e outro vai me devolver “Miau!” sendo resultados diferentes para a mesma chamada, fazendo assim ser polimorfismo.

**Importância**

É claro a importância da programação orientada a objeto e as duas das suas principais características, polimorfismo e herança, além de economizar muito código e facilitar a vida do programador quando está fazendo códigos mais extensos e complexos, ele facilita no tratamento de erros, afinal ficará mais fácil de identificar em qual método o erro está acontecendo e como consertar isso, melhora a visualização do código por parte de outros programadores e pessoas, e também auxilia a tornar um projeto escalável considerando que você vai poder herdar atributos de outras classes, sem precisar ficar escrevendo novas variáveis cada um para um elemento etc, além de também conseguir utilizar um método só para várias classes diferentes cada um dando sua resposta e aplicação específica com o polimorfismo, auxiliando na organização e economia do código.

**Fonte de pesquisa:**

<https://www.estrategiaconcursos.com.br/blog/programacao-orientada/>

<https://dev.to/fabianoflorentino/programacao-orientada-a-objetos-heranca-1pc3>

<https://blog.grancursosonline.com.br/heranca-na-programacao-orientada-a-objetos/>

<https://www.devmedia.com.br/conceitos-e-exemplos-heranca-programacao-orientada-a-objetos-parte-1/18579>

<https://dev.to/fabianoflorentino/programacao-orientada-a-objetos-polimorfismo-230b>

<https://blog.grancursosonline.com.br/poo-principal-conceito-de-polimorfismo/>

<https://www.devmedia.com.br/conceitos-e-exemplos-polimorfismo-programacao-orientada-a-objetos/18701>