Programação Orientada a Objetos (POO):

A Programação Orientada a Objetos (POO) nada mais é do que um padrão de desenvolvimento de programação. Ela organiza o código do programa em torno de objetos, que são a representação de coisas do mundo real, e esses objetos possuem características (atributos) e comportamentos (métodos).

Na POO, um objeto é uma instância de uma classe. Sendo que a classe é como se fosse um molde, que representa uma estrutura que define as características e comportamentos que um objeto pode ter.

Para entender melhor esse conceito imagine um cachorro, ele apresenta várias características e comportamentos, dessa forma, colocando essa situação em uma programação ela se daria com a criação de uma classe chamada "cachorro", onde ela possuí algumas características/ atributos (nome, raça, cor, gênero, etc.) bem como alguns comportamentos/ métodos (latir, comer, dormir, passear, etc.).

A partir daí, haveria a possibilidade de criar objetos nessa classe cachorro com valores específicos de cada atributo. A exemplo disso, dentro da classe "cachorro" eu poderia criar um objeto "Vira-lata" com atributos diferentes do objeto "Pitbull", mas ambos ainda estariam na classe "cachorro".

Além disso, a POO apresenta quatro pilares principais que são importantes para a implementação correta da programação.

Veja a baixo quais são eles:

Foto: Pilares da Programação Orientada a Objeto

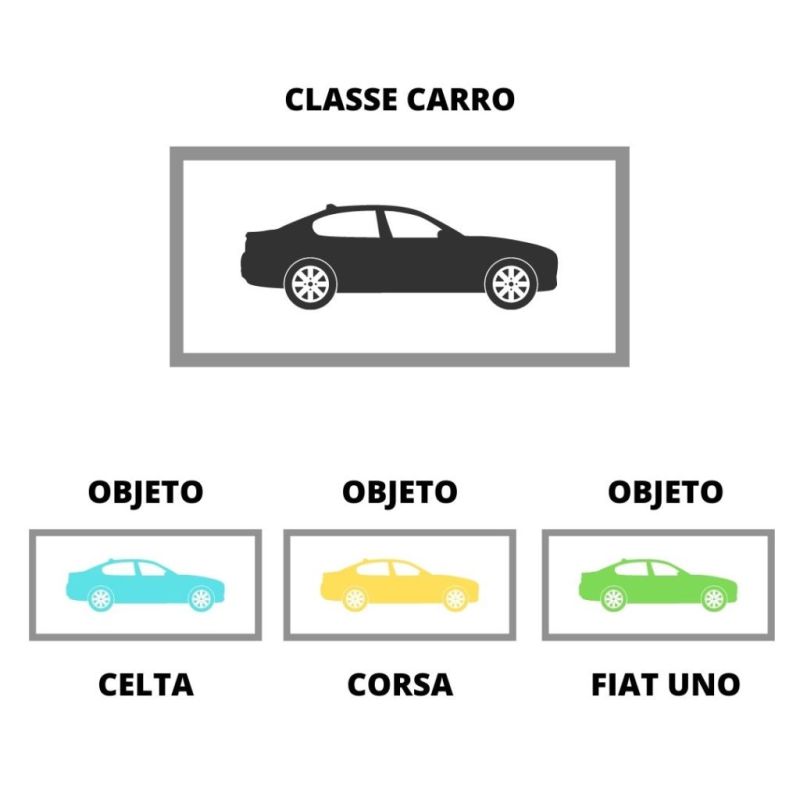
<https://lyncas.net/wp-content/uploads/2023/04/pilares-programacao-orientada-a-objetos.jpg>

Abstração:

Abstração é esconder na programação detalhes complexos a fim de expor o essencial na interface para o usuário.

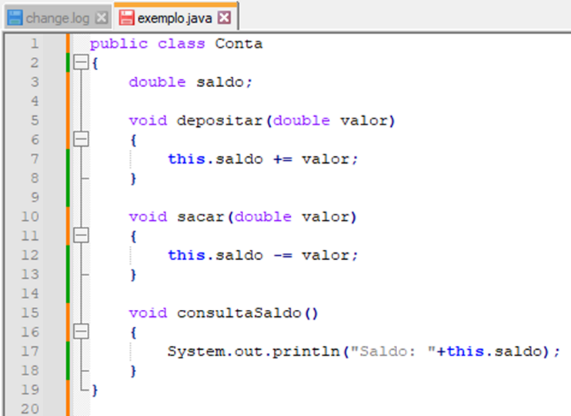
Sendo assim, é possível eliminar detalhes desnecessários, reduzir a complexidade e os impactos de mudanças feitas no código além de deixar uma interface simples e amigável.

Analisando mais uma vez o exemplo do cachorro, você não precisa saber todo o processo de digestão do animal para ele executar o método "comer" assim, na programação é possível eliminar detalhes que não interferem na compreensão da ação em si, tornando o programa mais simples ao usuário.

Foto: Exemplificação

https://hermes.dio.me/articles/cover/923dc46b-b122-444a-820d-699310434a7c.jpg

Segue a baixo um exemplo de Abstração no Java:

Foto: Exemplo Programação em Java de Abstração

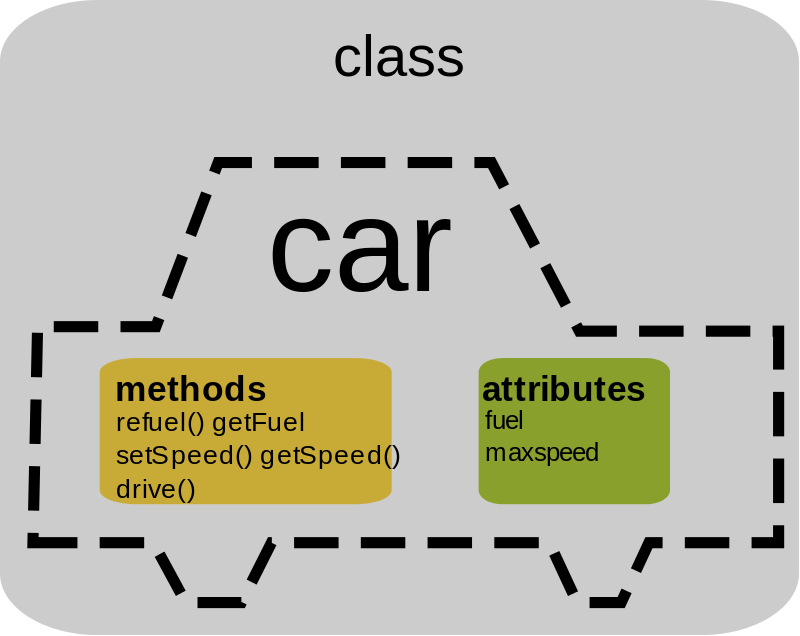
<https://dhg1h5j42swfq.cloudfront.net/2023/07/22122018/image-312.png>

Encapsulamento:

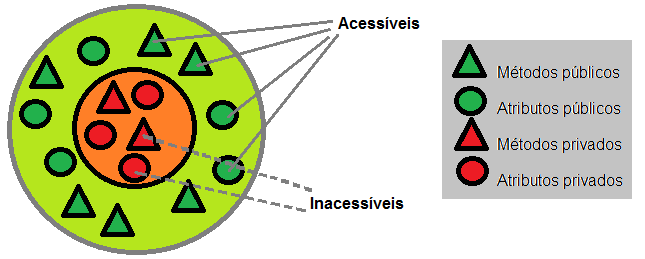
O encapsulamento é o processo de agrupar atributos e métodos relacionados dentro de um objeto, tornando-os “encapsulados”, como o nome sugere.

Isto permite que o objeto controle o acesso aos seus atributos e métodos, definindo quais são públicos (acessíveis a outros objetos) e quais são privados (acessíveis ao próprio objeto), além de promover a proteção e reduzir a complexidade, é possível reutilizar os objetos, possibilitando a reusabilidade.

Na nossa programação exemplo consiste em colocar na classe "cachorro" os atributos, como "raça", "nome", "gênero", etc., além dos métodos como "latir", "correr", "comer", etc.

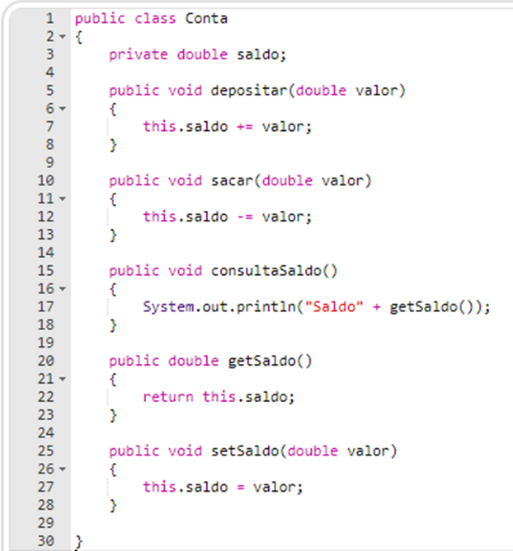
 Foto: Exemplificação Encapsulamento

<https://www.alura.com.br/artigos/assets/poo-programacao-orientada-a-objetos/oop-car-methods-and-attributes.png>

 Foto: Exemplificação Encapsulamento

<https://www.alura.com.br/artigos/assets/poo-programacao-orientada-a-objetos/encapsulation.png>

Segue a baixo um exemplo de Encapsulamento no Java:

 Foto: Exemplo Programação em Java de Encapsulamento

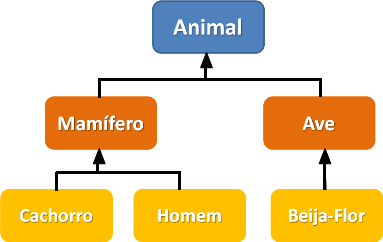
<https://dhg1h5j42swfq.cloudfront.net/2023/07/22122630/image-313.png>

Herança:

A herança é um mecanismo que possibilita a uma classe herdar atributos e métodos de outra classe. Sendo assim, é possível criar novas classes baseadas em classes já existentes. A classe que está sendo herdada é chamada de classe base ou superclasse, enquanto a classe que herda é chamada de classe derivada ou subclasse.

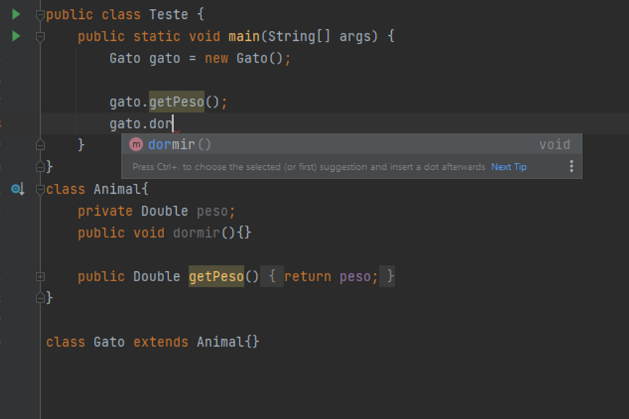
Com isso, esse método permite a reutilização de código e a criação de hierarquias de classes, eliminando repetições de código.

Digamos, dessa forma, que antes de termos feito a classe chamada "cachorro" nós tivéssemos feito uma classe chamada "animais", onde há métodos que se repetem na outra classe, tais como "comer", "beber", "respirar", etc. Sendo assim, para evitar escrevermos duas vezes esses mesmos métodos nas duas classes, é possível tornar a classe "animal" a classe base e a "cachorro" a classe derivada que vai herdar esses dados sem necessidade de repetição.

 Foto: Exemplificação Herança

<https://www.alura.com.br/artigos/assets/poo-programacao-orientada-a-objetos/inheritance.png>

Segue a baixo um exemplo de Herança em Java:

 Foto: Exemplo Programação em Java Herança

<https://blog.formacao.dev/content/images/size/w1000/2023/12/4-2.png>

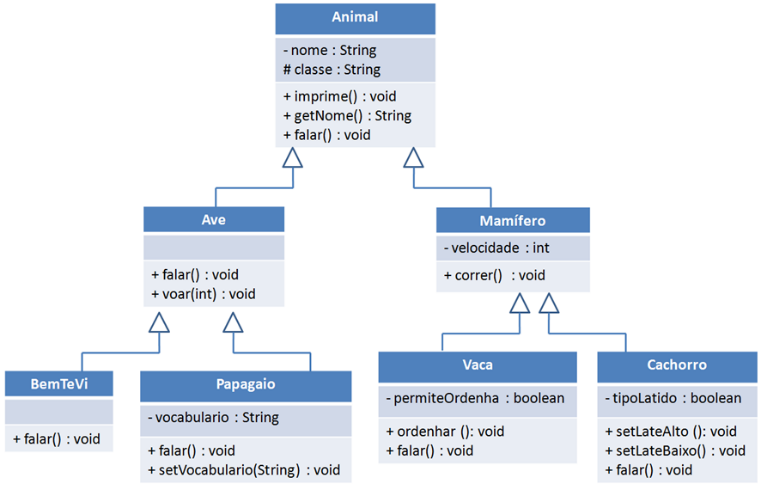
Polimorfismo:

"Poli" significa muitas e "Morphos" significa forma, então Polimorfismo significa muitas formas.

Em POO Polimorfismo é caracterizado quando duas ou mais classes possuem métodos com o mesmo nome, mas podendo ter implementações diferentes. Dessa forma é possível que diferentes objetos respondam a um mesmo método de maneira específica para cada um.

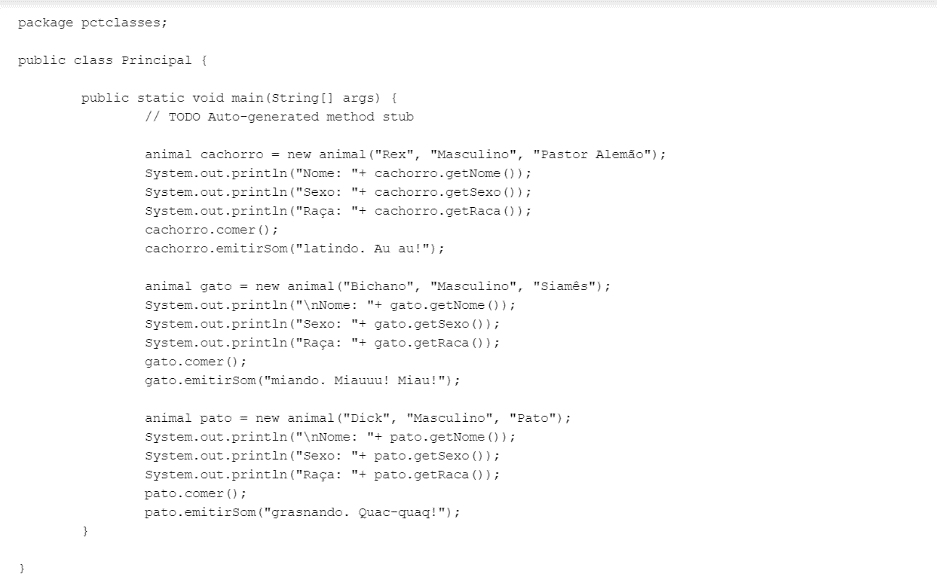
Com isso nós podemos remover if/ switch statements, simplificando o código e aumentando a flexibilidade do sistema.

Voltando para a analogia do cachorro, temos a classe "cachorro" e dentro dela o "Vira-lata" e o "Pitbull". Ambos podem executar o método “latir”, entretanto, cada um deles pode fazer de uma maneira diferente esse mesmo método.

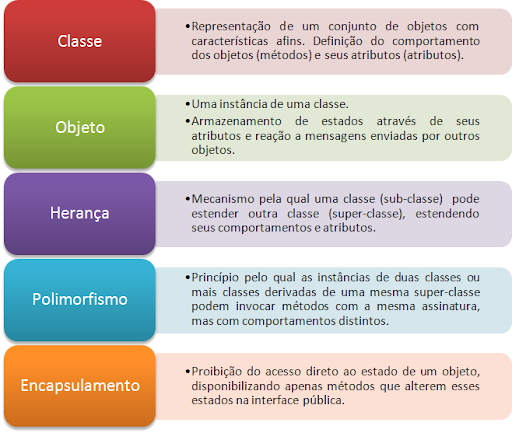
 Foto: Exemplificação Polimorfismo

<https://blogger.googleusercontent.com/img/b/R29vZ2xl/AVvXsEg9bsRnNtfmz_FvlG6aw2I1BlJpj43RGKisepIPSB-ws7I7GyUjq3laX-Ow6nwXLI2ZHcRT7WSGtVwxfl8o4ahrZAPT_GmM3yVU9pdBNjXtmsNZ3vL9NaG-NRko9hMxeSG1zJQwK6Y4uUmJ/s640/diagrama-fonte-cafw.ufsm.png>

Segue a baixo um exemplo de Polimorfismo no Java:

 Foto: Exemplo Programação em Java Polimorfismo

<https://www.canva.com/design/DAGNlfwlnfs/DtZTWsuqa0H_aBBeESUVdg/edit?utm_content=DAGNlfwlnfs&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton>

 Foto: Pilares da POO

https://lh3.googleusercontent.com/proxy/jJO7qL0yJSbv63iqbrrtVn5MEIutZCd5NMCm-KfKEeLu5xpR\_nqbQxAZdx7wImqjfuOj1k0tZ0yyVvVMLQT52o5MqW1m4V33w5ad-HscUZvLvZEzJHVAfmJfNXElsXromplnxpu427jk\_yKTY4C12iOWzhxXiSELEio\_\_xh0iaaZCVaKfQ

Conclusão:

Assim sendo, foi possível aprender mais acerca da Programação Orientada a Objeto. Foi possível observar que ela é uma estrutura de programação que consiste em objetos e classes, que para um bom funcionamento necessitam de 4 pilares

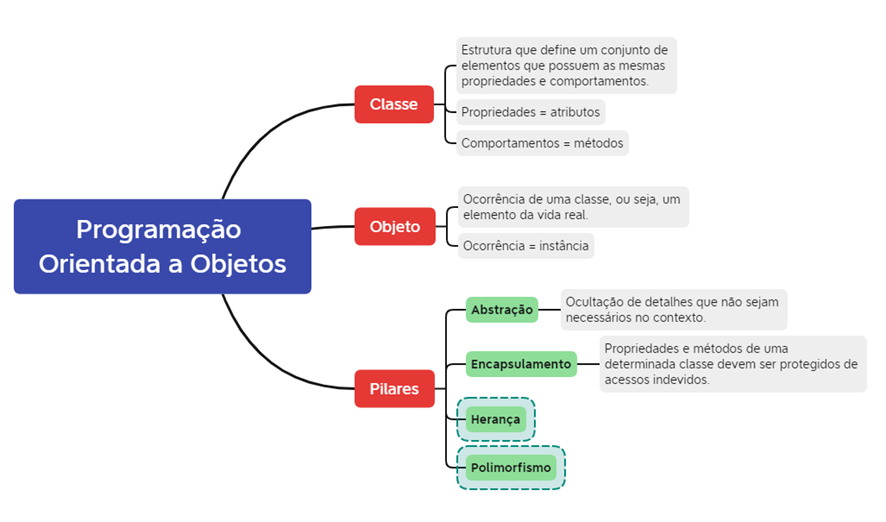
Existe a Abstração, onde detalhes não serão mostrados para a interface que o usuário usará, contribuindo para a simplificação do programa.

Além disso há o Encapsulamento, em que atributos e métodos são agrupados dentro de um objeto, sendo possível definir o acesso aos dados para certas pessoas, ajudando na segurança e complexidade.

Continuando com a Herança, vimos que ela é o método para uma classe herdar métodos e atributos de uma outra classe, colocando-a como classe base, facilitando o programa sem necessidade de repetições.

Por último possui o Polimorfismo, que possibilita que objetos de uma mesma classe executem um mesmo método de forma específica para cada um.

Com isso concluímos o assunto 😉

 Foto: Esquema sobre POO

<https://dhg1h5j42swfq.cloudfront.net/2023/07/22122822/image-314.png>

Perguntas Frequentes:

O que é uma classe? É uma estrutura que define um conjunto de elementos com as mesmas características e comportamentos.

O que é um objeto? É a ocorrência de uma classe que corresponde a um elemento da vida real, essa ocorrência também pode ser chamada de instância.

O que são os atributos? São as características de um objeto.

O que são os métodos? São comportamentos e ações que caracterizam um objeto.

O que é a Abstração? É um pilar da POO em que partes da programação que não afetam o entendimento do usuário serão escondidas para ela ser mais simples.

O que é o Encapsulamento? Mais um pilar da POO, onde os atributos e métodos vão ser agrupados em um objeto.

O que é Herança? Um dos pilares da POO, consiste em agrupar duas ou mais classes para uma herdar os métodos e atributos da outra, evitando repetições na programação.

O que é Polimorfismo? Outro pilar da POO, permite que um método herdado seja executado de maneira diferente entre os objetos.

E se ainda restou alguma dúvida?

Caso ainda tenha alguma dúvida, não deixe de conferir as fontes e assistir os vídeos sobre o tema 😊

Segue a baixo as fontes:

Fontes:

* https://www.dio.me/articles/os-4-pilares-da-programacao-orientada-a-objetos-Y0CN7G
* https://awari.com.br/os-4-pilares-da-programacao-orientada-a-objetos-guia-completo-para-iniciantes/
* https://www.treinaweb.com.br/blog/os-pilares-da-orientacao-a-objetos
* https://www.estrategiaconcursos.com.br/blog/programacao-orientada/
* https://www.alura.com.br/artigos/poo-programacao-orientada-a-objetos
* <https://www.youtube.com/watch?v=dXZRgW-X2ls>
* <https://www.youtube.com/watch?v=x9aLELLVzjQ>

Link do HTML:

<file:///C:/Users/Desenvolvedor/Documents/top%2010/site.html>