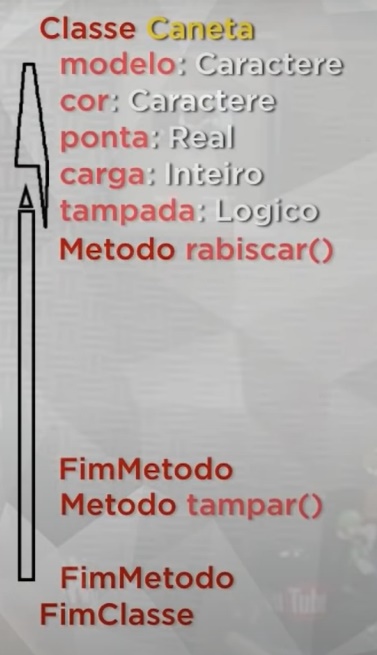
**Objeto:** Coisa material ou abstrata que pode ser percebida pelos sentidos e descrita por meio das suas características, comportamento e estado atual. (É a instancia de uma classe)

**Classe:** a classe de um objeto no POO deve responder a três perguntas, “Coisa que eu tenho”, “Coisa que eu faço”, “Como estou agora”. (Define atributos e métodos comuns que serão compartilhados por um objeto)

Onde **Coisa que eu tenho:** Atributo

**Coisa que eu faço:** Método

**Como estou agora:** Estado

**Ex:**

**Instanciar:** Quando se tem uma classe e gera-se um objeto a partir dela.

Conceito de **Abstração:** São atributos que não se tem importância para o Objeto.

**“this.”objeto”**: Sempre que for modificar um atributo dentro da própria classe.

**+ é público (public):** A classe atual e todas as outras classes podem acessar.

**- é privado (private):** Somente a classe atual pode acessar.

**# é protegido (protected):** Classe atual e todas as suas sub-classes.

**Getters:** Conseguem acessar um determinado atributo mantendo a segurança do mesmo.

**Setters:** Conseguem modificar um determinado atributo mantendo a segurança do mesmo.

**Construtor**: Método que deixa já pré-determinado um objeto com seus atributos. (Ele tem o mesmo nome da classe)

**PILARES DA PROGRAMAÇÃO ORIENTADA A OBJETO**

**Encapsulamento:** Ocultar partes independentes da implementação, permitindo construir partes invisíveis ao mundo exterior.

- Interfaces não tem atributos, apenas métodos.

**Herança:** Permite basear uma nova classe na definição de uma outra classe previamente existente. (A herança será aplicada tanto para as características quanto aos comportamentos)

**Polimorfismo:** Permite que o mesmo nome represente vários comportamentos diferentes.

**Sobreposição:** Acontece quando substituímos um método de uma superclasse na sua subclasse, usando a mesma assinatura.

**Sobrecarga:** Varias formas de chamar um método, independente da assinatura, porem é feito na mesma classe

**Classe Abstrata:** Não pode ser instanciada. Só pode servir como progenitora

**Método Abstrato:** Declarado, mas não implementado na progenitora.

**Classe Final:** Não pode ser herdada por outra classe. Obrigatoriamente folha.

**Método Final:** Não pode ser sobrescrito pelas suas sub-classes. Obrigatoriamente herdado.

**Assinatura do Método:** É a quantidade e os tipos de parâmetros (Variáveis e tipos).