

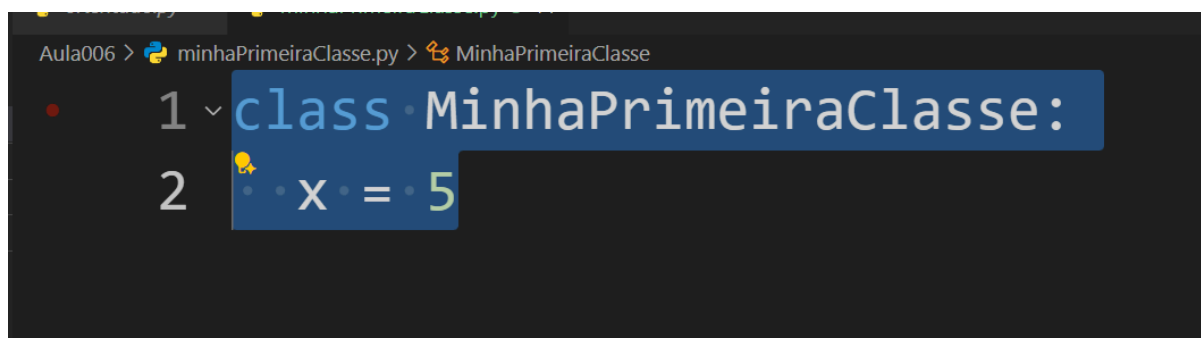
Python é Orientado a Objetos? SIM...

Uma classe em <QUAQUER LINGUAGEM> define um tipo de dado personalizado
e um objeto é uma instância dessa classe. VERDADE

— Em **Programação Orientada a Objetos (POO)**, uma **classe** é uma **estrutura que define um tipo de objeto** — ou seja, é um **molde** (ou **modelo**) a partir do qual os objetos são criados. —

Uma classe e um objeto são a mesma coisa em Python. NAOOOOOOO

O padrão do nome das classes é CamelCase.
SempreAPrimeiraLetraDaPalavraEEscritaEmMaiusculo

A screenshot of a Python IDE with a dark background. The top bar shows the file path 'Aula006 > minhaPrimeiraClasse.py' and the class name 'MinhaPrimeiraClasse'. The code editor shows two lines of Python code: '1 class MinhaPrimeiraClasse:' and '2 x = 5'. The first line is highlighted with a blue selection bar, and the second line is highlighted with a lighter blue selection bar. A small yellow icon is visible next to the second line.

```
Aula006 > minhaPrimeiraClasse.py > MinhaPrimeiraClasse
1 class MinhaPrimeiraClasse:
2     x = 5
```

Estrutura basica de uma classe...

Primeira linha:

CLASS -> Palavra reservada... Nao podemos utilizar pra mais nada essa palavra...

MinhaPrimeiraClasse -> É o nome da classe...

Quando vamos criar um objeto utilizamos

m = MinhaPrimeiraClasse()

- > utilizando uma função que existe em todas as classes chamada CONSTRUTOR....

TODA CLASSE TEM

- Nome
- Construtor
- Atributos
 - Dados que o objeto tem
 - propriedades dos objetos
 -
- Métodos
 - São coisas que a classe faz.
 - são métodos
 - Só não podemos chamar de funções...

Atributos PRIVADOS

COLOCAR dois _ antes do nome da propriedade

Ex. RENAVAN -> __renavam

```
self.__renavam = renavam
```

```
class Carro:          # Classe
    cor = ""          # Atributo
    modelo = 0         # Atributo
    ano = 0            # Atributo
    tipoRoda = ""      # Atributo
    __renavam = ""     # Atributo PRIVADO, somente a classe tem acesso
    __tank = 0         # Atributo PRIVADO, somente a classe tem acesso

    def __init__(self, cor, mod, ano, tipoRoda, renavam): # Construtor com parametros
        self.cor = cor
        self.modelo = mod
        self.ano = ano
        self.tipoRoda = tipoRoda
        self.__renavam = renavam

    def mostrarDetalhes(self): # Metodo, sem parametros, e SEM retorno
        print("Eu sou um carro")
        print("Cor: " + self.cor )
        print("Ano: " + str( self.ano))
        print("Modelo: " + str( self.modelo))

    def lerRenavam(self): # Metodo, sem parametros, e COM retorno
        return self.__renavam

    def abastecer(self, quantidade): # Metodo, COM parametros, e SEM retorno
        self.__tank = self.__tank + quantidade

    def painelGasolina(self): # Metodo, SEM parametros, e COM retorno
        return self.__tank
```

O construtor é definido pela função INIT

```
def __init__
```

Pode receber parâmetros, (ex. COR, MOD, ANO)

```
1 class Carro:
2     cor = ""
3     modelo = 0
4     ano = 0
5
6     def __init__(self, cor, mod, ano):
7         self.cor = cor
8         self.modelo = mod
9         self.ano = ano
10
```

Documentos sem título - Docu... x Welcome To The Python Tutor... x Python Class Methods x +

w3schools.com/python/python_class_methods.asp

W3schools Tutorials References Exercises Certificates Search...

HTML CSS JAVASCRIPT SQL PYTHON JAVA PHP HOW TO W3.CSS C C++ C# BO

Python Try...Except
Python String Formatting
Python None
Python User Input
Python VirtualEnv
Python Classes
Python OOP
Python Classes/Objects
Python __init__ Method
Python self Parameter
Python Class Properties
Python Class Methods
Python Inheritance
Python Polymorphism
Python Encapsulation
Python Inner Classes
File Handling

Example

Create a method in a class:

```
class Person:
    def __init__(self, name):
        self.name = name

    def greet(self):
        print("Hello, my name is " + self.name)

p1 = Person("Emil")
p1.greet()
```

[Try it Yourself »](#)

Note: All methods must have `self` as the first parameter.

CONTEUDO sobre Orientação a Objetos com Python

Turma

Paradigmas de Linguagens de Programação em Python

[Início](#) [Cont. Complementar](#) [Trabalhos](#)

Plano de Aula

Baixar

Plano de Ensino

Baixar

Tema 1

Introdução

>

Tema 2

Paradigmas e Linguagem Python

>

Tema 3

Python Básico

>

Tema 4

Python Estruturado

>

Tema 5

Python Orientado a Objetos

>

Tema 6

Prepare

>

Tema 5

Python Orientado a Objetos

Python Orientado a Objetos

13% visualizaram

Conteúdos Complementares (1)

Você pode adicionar mais conteúdos complementares a este tema.

Adicionar Conteúdo

Repositorio Git

link tema 1,2,3,4,5,6 visualizado

49% visualizaram

Conteudos importantes

https://www.w3schools.com/python/python_conditions.asp

https://www.w3schools.com/python/python_while_loops.asp

https://www.w3schools.com/python/python_for_loops.asp

https://www.w3schools.com/python/python_lists.asp

https://www.w3schools.com/python/python_output.asp

https://www.w3schools.com/python/python_user_input.asp

PARA PRVA FINAL

Tudo mais estes abaixo

https://www.w3schools.com/python/python_pip.asp

[https://www.w3schools.com/python/python_try_except.a
sp](https://www.w3schools.com/python/python_try_except.asp)

https://www.w3schools.com/python/module_math.asp