

## POO – Programação Orientada a Objetos

Objetivo: revisão e imersão da orientação a objetos

No projeto da API crie uma pasta POO (programação orientada a objetos)

### O que é uma classe?

R: Um conjunto de objetos

### Para que serve uma classe?

Para categorização ou agrupamento de objetos com características e ou funcionalidades semelhantes.

### Como definir uma classe?

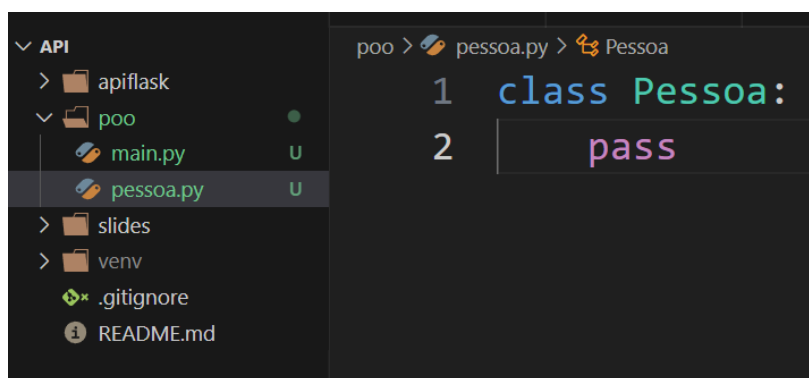


Figura 1 - Não esqueça que toda classe começa com letra maiúscula.

### Exercício:

Analisando o contexto dos gastos financeiros da sua família. Crie uma classe chamada **conta**.

### O que é um construtor?

Um método especial para construir o objeto na memória.

Enquanto o construtor não é chamado, a variável é apenas uma referência.

```
def __init__(self):
    pass
```

### Atributos

São as características dos objetos.

```
def __init__(self, descricao, valor, id=None):
    self.id = id
    self.descricao = descricao
    self.valor = valor
```

Essas características no Python devem ser definidas apenas dentro do construtor, usando a palavra reservada **self**.

## Diário de Bordo

<b>Categoria</b>	<b>Título</b>
Situação-problema	Classe, Construtor, Atributo e Objeto
1363 Caracteres restantes	
<b>Descrição da Situação de Aprendizagem</b>	
<p>Classe é um conjunto de objetos O objeto é a instância de uma classe Na classe define-se os métodos ou funcionalidades dos objetos O construtor serve para iniciar os atributos da classe e criar o objeto em memória.</p>	
3784 Caracteres restantes	
<b>Recursos Didáticos *</b>	
<p><a href="https://github.com/romulosilvestre/apiflask/blob/master/poo/docs/conte%C3%BAdo.png">https://github.com/romulosilvestre/apiflask/blob/master/poo/docs/conte%C3%BAdo.png</a></p>	
2318 Caracteres restantes	

## Conteúdo Geral

<https://github.com/romulosilvestre/apiflask/blob/master/poo/docs/conte%C3%BAdo.png>

## Código-Fonte

<https://github.com/romulosilvestre/apiflask/blob/master/poo/pessoa.py>

The screenshot shows a GitHub repository interface. On the left, the file explorer shows the directory structure: 'apiflask' (folder), 'poo' (folder), 'docs' (folder), 'conteúdo.png' (file), 'conta.py' (file), 'main.py' (file), 'pessoa.py' (file), 'slides' (folder), and '.gitignore' (file). The 'pessoa.py' file is selected. On the right, the code editor shows the content of 'pessoa.py' with 11 lines of Python code. A red arrow points from the 'pessoa.py' file in the file explorer to the code editor.

```
1 class Pessoa:
2
3     def __init__(self, nome, idade, cpf=None):
4         self.nome = nome
5         self.idade = idade
6         self.cpf = cpf
7
8
9 if __name__ == "__main__":
10     pessoa = Pessoa()
11     pessoa.nome = "Joaquim da Silva"
```

Listagem de código:

```
class Pessoa:
```

```
def __init__(self,nome,idade,cpf=None):
    self.nome = nome
    self.idade = idade
    self.cpf = cpf

if __name__ == "__main__":
    pessoa = Pessoa()
    pessoa.nome = "Joaquim da Silva"
```

Expliquei o uso do **self**.

O self (lembra aquele self para tirar foto).

Ele define o que a classe tem.

Nesse caso ela é responsável por definir os atributos da classe.

