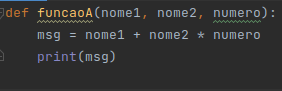
**52 – Funçõe (def) em Python**

Definindo funções em python

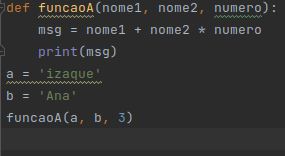
Sintaxe:

**def <name>(<paramet1>, ... , <parametN>):**

**<bloco da função>**

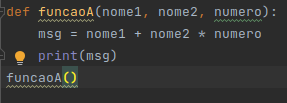


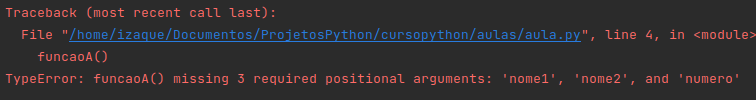
Vamos chamar essa função:

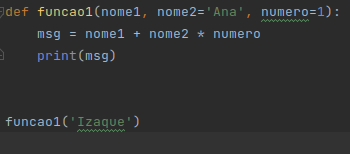
Podemos colocar valores padrão para os parâmetros da função, eles terão esse valor padrão caso dermos um argumento para eles quando chamarmos a função.

Por exemplo, se não dermos valores padrão aos parâmetros e chamarmos a função sem parâmetros ocorrerá um erro:





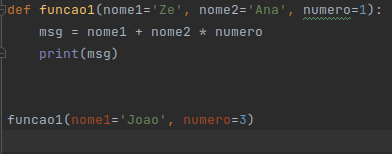
Vamos agora dar valores padrão aos parâmetros, basta colocar o sinal de atribuição =, e o valor padrão:

Veja que não foi dado nenhum argumento aos parâmetros nome2 e numero, então eles assumirão seus valores padrão.

Obs: Os parâmetros que possuem valor padrão devem ser os últimos na lista de parâmetros da função.

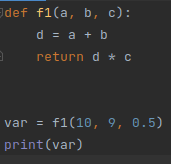
Também podemos chamar uma função, dando os argumentos de forma nomeada, veja um exemplo:

Obs: Os argumentos nomeados devem ser os últimos na lista de argumentos da função.

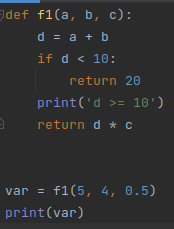
Funções como já sabemos podem retornar valores.

Para isso utilize o **return <valor>**, e no momento que ela chegar nessa linha de código a função irá terminar e irá retornar esse <valor>. Veja um exemplo:

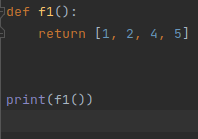
 

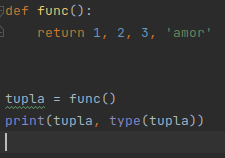
Por curiosidade uma função que não possui **return** no seu corpo retorna o dado **None**.

É importante destacar também que se caso a linha de código que está o **return** for executada, a função irá terminar, e toda linha de código da função que está abaixo de return, não será executada.

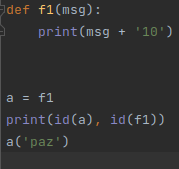
 

Uma função não retorna só dados primitivos, ela pode retornar objetos iteráveis por exemplo:

Variáveis podem apontar para uma função:

(Obs: id(<object>) retorna o endereço da memoria de um certo dado

<https://docs.python.org/3/library/functions.html#id> )

Entenda também a diferênça, ao fazer:

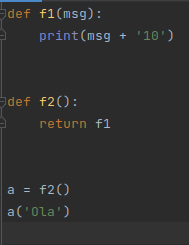
<var> = <function>

<var> passa a apontar para a função <function> mas a função não é executada, ao fazer:

<var> = <function>()

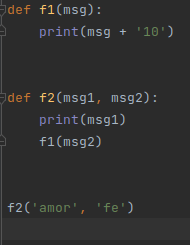
A função <function> é executada, e o valor que ela retorna é atribuido à <var>

Aproveitando o mencionado, funções podem retornar funções.

a recebeu o valor que a função f2 retorna, que é a função f1, depois é chamada a função f1 por meio do a com o argumento ‘Ola’

Claro que também podemos chamar funções dentro de outra:

Assim como também funções podem receber funções como parâmetros:

