Varios argumentos xargs

Já vimos como fazemos para passar **parâmetros** em uma **função** certo? Como por exemplo...

```python
def somar_dois_numeros(numero1: float, numero2: float):
    resultado = numero1 + numero2
    print(f'{resultado:.2f}')
```

Function com MULTI parâmetros (xargs)

Mas e se não quisermos criar uma função com **parâmetros limitados**, ou seja, queremos passar como fosse **parâmetros infinitos**!! Até porque tem casos que podemos deixar flexível para quem for consumir a nossa função, possa passar a quantidade de argumentos que quiserem!!

E para isso utilizamos o recurso chamado **argumentos xargs**!! E para exemplificar vamos criar uma função de **soma**, onde podemos somar **N** números que quisermos e não só apenas 2 números.

03 - Funcoes > 03 - Varios argumentos xargs > codigo-fonte.py > ...

```python
# Function com parametro xargs(*)
def somar_numeros(*numeros):
    resultado = 0

    for num in numeros:
        resultado += num

    print(resultado)


somar_numeros(10, 20, 30)

```

Vale destacar que o **xargs** do **python** refere há um objeto do tipo **Tuple<>**, e como nesse objeto **implementa** o **Iterable**, logo conseguimos utilizar ele no **FOR**. Até porque, como já vimos, internamente o **FOR** do **python**, chama o método **next()**, para chamar a sequência da sua iteração!

Function com MULTI parâmetros kwargs (**)

Vamos agora pensar em um cenário diferente, onde seria importante ao passar os argumentos da função dos **parâmetros xargs**, mas definimos o nome de cada parâmetro que queremos passar.

```python
somar_numeros(numero1=10, numero2=20, numero3=30)
```

```
raceback (most recent call last):
 File "c:\Projetos\pocs\python-study\03 - Funcoes\03 - Varios argumentos xargs\codigo-fonte.py", line 14, in <module>
   somar_numeros(numero1=10, numero2=20, numero3=30)
ypeError: somar_numeros() got an unexpected keyword argument 'numero1'
```

Veja que teremos um erro acima!

E isso acontece porque quando definimos os nossos parâmetros **xargs** somente com um ***** estamos definindo que podemos passar **N** argumentos, porém não estará esperando **"nomes fixos"** de **parâmetros** nele!

Mas para resolver isso é muito simples!! Ao invés de utilizar o **xargs(*)** usaremos o **kwargs(**)** como parâmetro na **function**!

E para exemplificar direitinho, vamos cria ruma nova função, onde irá ser listado todas as informações de um carro, através dos **parâmetros kwargs(**)**!

```python
def mostar_info_carro(**carro):
    for carroKey, carroValue in carro.items():
        print(f'{carroKey} = {carroValue}')


mostar_info_carro(marca='jeep', cor='azul', motor=2.0)
```

```
marca = jeep
cor = azul
motor = 2.0
```

Veja que ao utilizar os **parametros kwargs(**)** ele funciona como um **Dictionary**, ou seja, trabalharemos com **chave** e **valor** nele.

Resumo xargs(*) VS kwargs(**)

Para finalizar o entendimento, de uma forma bem resumida as principais diferenças de parâmetros **xargs** e parâmetros **kwargs** são:

- **parametros xargs**: Usados quando queremos passar **múltiplos argumentos posicionais**. Eles são recebidos como uma **tupla**, funcionando como uma lista de valores.
  👉 Ideal quando **a ordem importa**, mas **os nomes não são necessários**.

- **parametros kwargs**: Usados quando queremos passar **múltiplos argumentos nomeados** (chave=valor). Eles são recebidos como um **dicionário**.
  👉 Ideal quando **os nomes (chaves)** são importantes, como fosse propriedades de um objeto.