Documentação

Dentro do Python, como forma de auxilio principalmente com relação a explicação de funções bem como oque um determinado módulo ou trecho do código faz é que construímos DocString. Para declarar uma DocString primeiro inserimos um conjunto de `"""..."""` ou com junto de três aspas (Poder ser simples ou dupla)

```
def soma (valor1: int, valor2: int) -> int:
    """
    Conteudo de uma DocString
    """
```

> Os DocString ajudam você a entender os recursos de um módulo ou de uma função. Por exemplo, digamos que você tenha instalado a biblioteca `scikit-learn` e queira saber tudo sobre o pacote `sklearn`, como descrição, módulos do pacote etc., você pode simplesmente usar a função `help` para obter todas as informações.

```
help(sklearn)
```

Sendo assim, devemos destacar as seguintes informações:

- Oque aquela função/trecho do código faz (Sempre devemos ser breves porém completos)
- Quais argumentos ele precisa
- Qual seu retorno

Caso tenha que documentar uma classe, devemos documentar sempre a classe como um todo e depois vim documentando cada método da mesma

```
"""
Nesse primeiro momento é definido de forma breve qual o intuito da função, c
qual sua finalidade
EX:
Essa função recebe dois valores do tipo int e retorna a soma desse números
"""
```

👉 Como boa prática, informado pela PEP 257, devemos sempre optar por passar os comandos no modo imperativo

---

Documentação em uma linha:

Como o próprio nome já menciona é uma breve documentação, geralmente destinadas a funções simples ou mesmo quando o nome das mesma é auto explicativo - Em resumo, é usada em casos óbvios

- Não devemos replicar seus parâmetros ou mesmo a sintaxe da função na sua documentação, no máximo informa seu retorno e qual tipo o mesmo será

```
def square(a):
    '''Returned argument a is squared.'''
    return a**a+
```

Documentação multilinhas:

Aplicada em casos maiores, onde temos funções mais complexas, classes, métodos e módulos, e sempre é resumido de forma breve sua funcionalidade seguido por quais argumentos são esperados ao chamar a classe, além de mencionar seu tipo (`int, float, str`) e por fim descrever qual o retorno esperado da função ou método que foi documentado

```
def aumento_salario (valor_salario: float)→ dicionario_retorno:
    """
        Função destinada ao cálculo do valor do salário com base na tabela de a
```

```
    15% para salários até R$1250,00 e 10% para salários acima de R$1250,0
    Informe o valor do salário e a função retornará um dicionário com as info

    Args:
        valor_salario (float): Valor do salário do funcionário
    Returns:
        dict: Dicionário contendo as informações do aumento
        - PORCENTAGEM_AUMENTO (10% ou 15%)
        - VALOR_AUMENTO qual o valor em R$ foi adicionado ao salário
        - SALARIO_FINAL qual o valor do salário após o aumento
"""
porcentagem_aumento = 0.10 if valor_salario >1250 else 0.15
aumento = valor_salario * porcentagem_aumento
valor_final = aumento + valor_salario
return{
    "PORCENTAGEM_AUMENTO": porcentagem_aumento*100,
    "VALOR_AUMENTO": aumento,
    "SALARIO_FINAL": valor_final
}
```

Uma documentação pode variar de necessidade para necessidade, ou de onde está sendo utilizada, um exemplo seria a diferença para documentar um módulo de uma documentação de uma função

Documentação de Função

Para documentar uma função, geralmente explicamos seu retorno, e sempre priorizando o modo imperativo, ou seja, faça isso e obtenha aquilo, porém de certa forma de forma mais simples e mais resumido

Documentação de Módulo

Como um módulo pode contar várias classes e com vários outros métodos e também pode contar funções, geralmente para documentar um módulo documentamos oque o mesmo pode fazer, quais são suas utilidades quais suas classes etc.

Documentação de Classe

Fornece uma documentação como um todo além de ser encontrada imediatamente depois da definição da classe

```
class MyClass:
    """This is the documentation for MyClass."""

    def __init__(self):
        """This is the documentation for the __init__ method."""
        pass
```

Para poder acessar uma documentação de uma classe por exemplo onde temos a DocString, podemos usar dois métodos

1. Método `help()` → Digitamos `help(...)` e passamos como parâmetro a classe função ou método que precisamos ver a documentação
2. Usando o `__doc__` → Quando queremos usar esse método digitamos o comando de `print(...__doc__)` onde os `"..."` é o nome da função, método ou classe que queremos acessar a documentação

De qualquer modo vamos ter acesso as informações destacadas a respeito da função que buscamos, **porém quando utilizamos o `help()` temos uma documentação mais detalhada sobre o mesmo,** principalmente se tratando de módulos que usamos, pois geralmente são classe com vários métodos que o compõe e quando utilizamos o `help()` ele retorna a descrição da classe como um todo e nome e descrição dos métodos que a compõe