

Funções

**Definindo funções
personalizadas em Python**

Prof. Me. Lucio Nunes

Prof. MSc. Rafael Maximo

Tópicos

Nesta aula iremos falar sobre:

- Funções importadas;
- Criação de funções em Python;
- Diferença entre retorno e exibição de valor;
- Escopo global e local;
- Texto de documentação (docstring); e
- Organização do código.

Objetivos

Acompanhe, a seguir, os objetivos de aprendizagem para esta unidade:

- Aplicar a importação de módulos;
- Entender como criar funções;
- Entender o conceito de escopo;
- Avaliar a diferença entre print e return; e
- Aprender sobre strings de documentação.

Importação de módulos

Usamos o comando `import` para carregar módulos extras do Python

```
>>> import math
>>> math.pi
3.141592653589793
>>> math.log(10)
2.302585092994046
```

```
>>> import time
>>> time.sleep(3)
>>>
```

Definindo funções em Python

Para definir uma função,
usamos a palavra chave def

```
def <nome da função>([<parâmetros>]):  
    <bloco de código da função>
```

Exemplo:

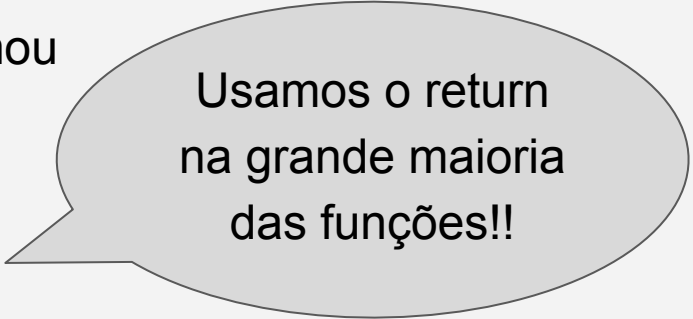
```
def somar(n1, n2):  
    soma = n1 + n2  
    return soma
```

A palavra chave
return encerra a
execução da função,
retornando o valor a
seguir, se houver.

print vs. return

Usamos return para encerrar a execução de uma função e devolver um valor ao código que a chamou

```
def somar(n1, n2):  
    soma = n1 + n2  
    return soma
```



Usamos o return
na grande maioria
das funções!!

Usamos print para simplesmente exibir um valor na tela, não interfere no fluxo e execução da função

```
def diga_ola(nome):  
    print(f'Olá {nome}!')
```

Escopo

Figura 1: Visualização da execução de uma função no Python Tutor (pythontutor.com)

The image shows the Python Tutor interface for Python 3.6. On the left, a code editor displays a function definition and its call:

```
Python 3.6  
(known limitations)  
  
1 def diga_ola(nome):  
2     print(f'Olá {nome}!')  
3  
4 nome = 'Megan'  
5 diga_ola(nome)
```

Below the code, a legend indicates that a green arrow points to the line just executed and a red arrow points to the next line to execute. A progress bar and navigation buttons (<< First, < Prev, Next >, Last >>) are also present, showing 'Step 5 of 6'.

On the right, the 'Print output' area is empty. Below it, the 'Frames' and 'Objects' panels are visible. The 'Frames' panel shows two frames:

- Global** (blue border): Contains variables `diga_ola` and `nome`. `nome` is assigned the value `"Megan"`.
- Escopo local** (red border): Contains variables `diga_ola` and `nome`. `nome` is assigned the value `"Megan"`.

An arrow points from the `diga_ola` variable in the Global frame to the `function diga_ola(nome)` object in the Objects panel. The text 'Escopo local' is written in red next to the local frame.

Escopo é o espaço no qual um identificador é válido e está associado a um valor

Fonte: elaborado pelo autor no site pythontutor.com

string de documentação

```
def somar(n1, n2):  
    """Retorna a soma de dois números  
  
    Parâmetros  
    -----  
    n1, n2: int ou float  
        Números a serem somados.  
  
    Retorno  
    -----  
    int ou float  
        Resultado da soma de `n1` com `n2`  
    """  
    soma = n1 + n2  
    return soma
```

Usamos a *docstring* para documentar a utilização da função, o que faz, quais parâmetros recebe e qual valor retorna, entre outros

Organizando nosso código

```
### Importações ###  
import math  
  
### Funções ###  
def somar(n1, n2):  
    return n1 + n2  
  
### Código principal ###  
a = int(input('Digite um número: '))  
soma = somar(a, math.pi)  
print(f'{a} + {math.pi:.3f} = {soma:.3f}')
```

A recomendação é dividir o código fonte em três partes:

1. Importações
2. Definição de funções
3. Código principal

Saiba +

Tutorial oficial da PSF
(python.org) sobre o
comando `def`

Guia do numpy sobre
docstrings



Referências

Docstring Guide. **Numpydoc** 2021. Disponível em: <<https://numpydoc.readthedocs.io/en/latest/format.html>>. Acesso em: 25 jan. 2021.

DOWNEY, A. B. **Pense em Python**. 1 ed. São Paulo: Novatec Editora Ltda., 2016.

PSF. **math: Mathematical Functions**. 2021. Disponível em: <<https://docs.python.org/3/library/math.html>>. Acesso em: 26 jan. 2021.

STURTZ, J. Defining Your Own Python Functions. **Real Python**, 2020. Disponível em: <<https://realpython.com/defining-your-own-python-function/>>. Acesso em: 25 jan. 2021.