

Computação I - Python

Aula 1 - Introdução à programação

Introdução ao uso de funções

Apresentado por: Carla A. D. M. Delgado

Produção DCC-UFRJ

Metodologia de referência <https://doi.org/10.5753/wei.2016.9683>



Funções em Python: definindo uma função

Exemplo: Vamos escrever a função *dobro* que, dado um número x , retorna o seu dobro, ou seja, $2 * x$:

Funções em Python: definindo uma função

Exemplo: Vamos escrever a função *dobro* que, dado um número x , retorna o seu dobro, ou seja, $2 * x$:

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código
- Ao esquema de posicionamento usado no Python damos o nome de *identação*. Ele é um recurso que faz parte da sintaxe da linguagem, e se estiver incorreto, seu código não vai funcionar :-)

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código
- Ao esquema de posicionamento usado no Python damos o nome de *identação*. Ele é um recurso que faz parte da sintaxe da linguagem, e se estiver incorreto, seu código não vai funcionar :-)
- No *cabeçalho* da função descrevemos as entradas com que a função vai trabalhar

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """ calcula e retorna o dobro do numero x de entrada """  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código
- Ao esquema de posicionamento usado no Python damos o nome de *identação*. Ele é um recurso que faz parte da sintaxe da linguagem, e se estiver incorreto, seu código não vai funcionar :-)
- No *cabeçalho* da função descrevemos as entradas com que a função vai trabalhar
- No *corpo* da função dizemos como ela é calculada

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """calcula e retorna o dobro do numero x de entrada"""  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código
- Ao esquema de posicionamento usado no Python damos o nome de *identação*. Ele é um recurso que faz parte da sintaxe da linguagem, e se estiver incorreto, seu código não vai funcionar :-)
- No *cabeçalho* da função descrevemos as entradas com que a função vai trabalhar
- No *corpo* da função dizemos como ela é calculada
- O primeiro item do corpo da função é sua documentação, que vem entre três aspas

Funções em Python: definindo uma função

```
1 def dobro(x):  
2     """calcula e retorna o dobro do numero x de entrada"""  
3     return 2*x
```

- A definição de uma função em Python começa com a palavra reservada *def*
- A estrutura da definição de uma função em Python tem duas partes: *cabeçalho* e *corpo*. Estas partes podem ser reconhecidas por seu posicionamento no código
- Ao esquema de posicionamento usado no Python damos o nome de *identação*. Ele é um recurso que faz parte da sintaxe da linguagem, e se estiver incorreto, seu código não vai funcionar :-)
- No *cabeçalho* da função descrevemos as entradas com que a função vai trabalhar
- No *corpo* da função dizemos como ela é calculada
- O primeiro item do corpo da função é sua documentação, que vem entre três aspas
- A palavra reservada *return* é usada para marcar o que deve ser retornado como resultado da função.

Funções em Python: definindo uma função

```
1 def nome_funcao(lista_parametros):
```

- Cabeçalho

- palavra reservada *def*
- nome da função, à sua escolha
- entradas da função, entre parênteses (), separadas por vírgula. O nome de cada entrada fica à sua escolha
- dois pontos : para indicar que a seguir virá o corpo da função

Funções em Python: definindo uma função

```
1 def nome_funcao(lista_parametros):
```

Como escolher nomes para funções e suas entradas?

- Por questões de legibilidade, é muito importante que você escolha nomes significativos, ou seja, que tenham a ver com o papel que aquele nome vai representar dentro da função.
- Existe também uma regra sintática da linguagem Python para formação de identificadores (nomes):
 - Começar por letra
 - Depois da primeira letra, você pode usar sequencias de letras, números e sublinha, mas não pode ter espaço no meio nem outros caracteres como acentos, #, @... **Só pode usar letra, número e _**
- Duas coisas não podem ter o mesmo nome em uma mesma função. Logo, nomes de função e entradas tem que ser diferentes.

Funções em Python: definindo uma função

```
1 def nome_funcao(lista_parametros):  
2     """Como se define uma funcao em Python"""  
3     return valor de retorno
```

- Corpo da função

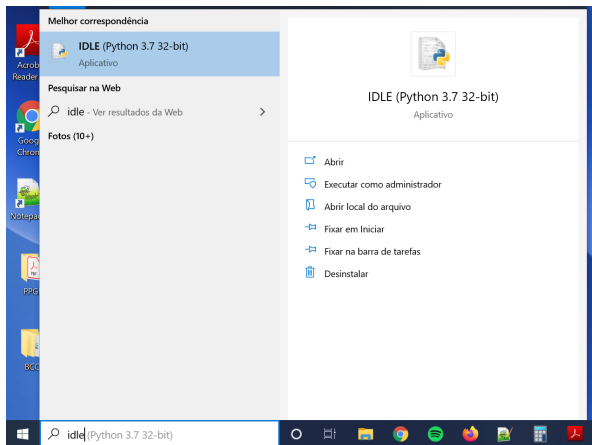
- Caracterizado sintaticamente pelo posicionamento *identado* à direita em relação ao cabeçalho.
- documentação: comentário entre três aspas descrevendo o que a função faz
- No corpo da função descrevemos como as entradas são usadas para gerar o resultado esperado da função, ou seja seu *valor de retorno*
- o comando **return** indica o que deve ser retornado.

Funções em Python: ferramenta de trabalho

- Vamos trabalhar com a ferramenta IDLE, que é instalada junto com o Python
 - IDLE: Integrated Development and Learning Environment
 - Para definir a função, usaremos o **editor** do IDLE
 - Para usar a função, usaremos o **Python Shell** do IDLE

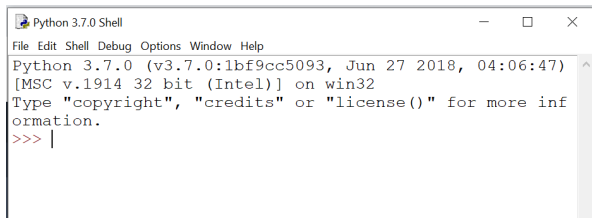
Abrindo o IDLE

Como rodar o IDLE? Busque pela palavra IDLE na barra de tarefas do sistema operacional.



Abrindo o IDLE - Python shell

Clique no IDLE e ele abrirá a janela com o shell do Python, ou seja, o interpretador rodando no modo interativo.

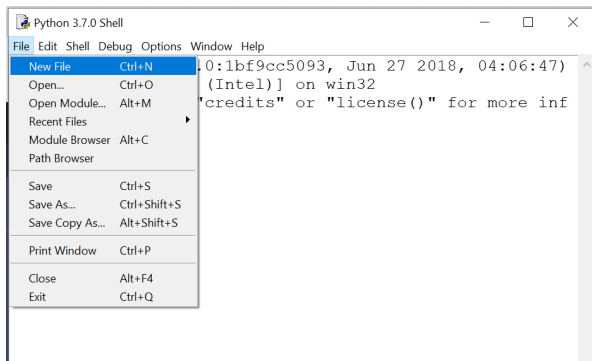
A screenshot of a Windows application window titled "Python 3.7.0 Shell". The window has a standard menu bar with "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main text area displays the following text: "Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32", followed by "Type 'copyright', 'credits' or 'license()' for more information." and a prompt ">>> |" on the next line. The window has standard minimize, maximize, and close buttons in the top right corner.

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47)
[MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more inf
ormation.
>>> |
```


Abrindo o editor do IDLE

Na janela do Python Shell, vamos abrir o **editor**:

'File > New File' ou **Ctrl + N**

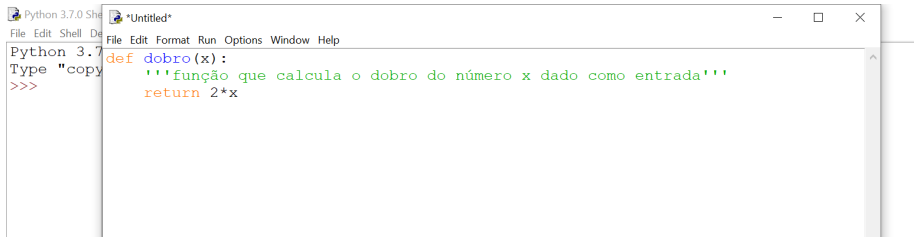


IDLE: escrevendo uma função no editor



Usaremos o editor do IDLE para editar nossas funções

IDLE: escrevendo uma função no editor

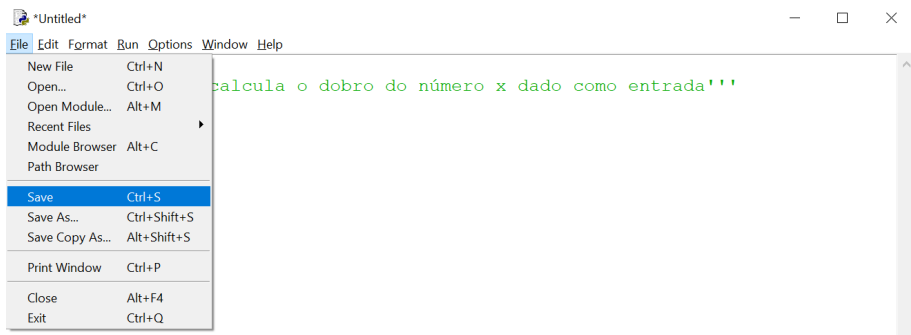


The screenshot shows the Python IDLE environment. On the left is the Python 3.7 Shell, and on the right is the *Untitled* editor window. The editor window contains the following Python code:

```
def dobro(x):  
    '''função que calcula o dobro do número x dado como entrada'''  
    return 2*x
```

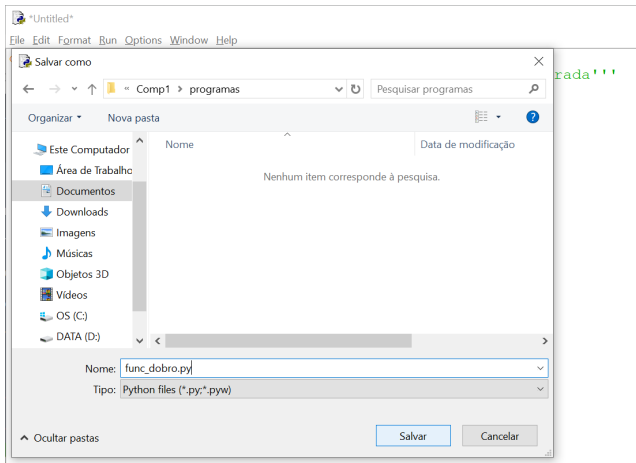
Escreva o código da função no editor do IDLE

IDLE: salvando seu código no editor



Guarde seu código em um arquivo:
selecione **'File > Save'** ou **Ctrl + S**

IDLE: salvando seu código no editor



Podemos usar o seguinte nome para este arquivo: **func_dobro.py**
É importante colocar .py após o nome do arquivo!

IDLE: Minha primeira função

Acabamos de escrever nossa função e salvamos o arquivo. :-)
Nosso código está pronto??

IDLE: Minha primeira função

Acabamos de escrever nossa função e salvamos o arquivo. :-)
Nosso código está pronto??

N Ã O !

Nenhum código pode ser considerado finalizado até que seja devidamente testado.

IDLE: Minha primeira função

Acabamos de escrever nossa função e salvamos o arquivo. :-)
Nosso código está pronto??

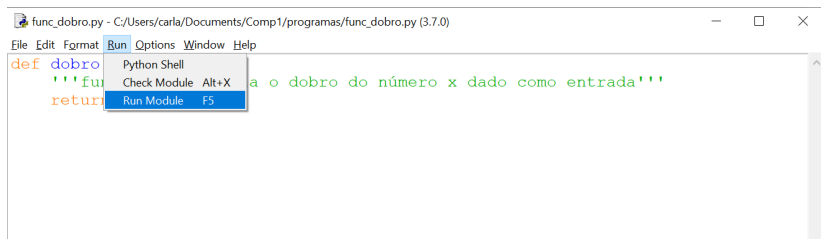
N Ã O !

Nenhum código pode ser considerado finalizado até que seja devidamente testado.

Para testar, temos que:

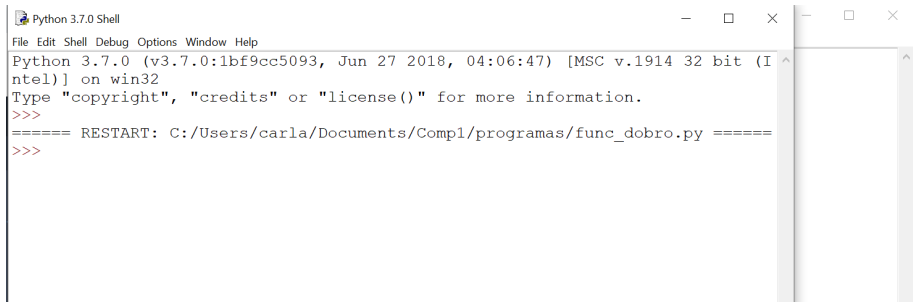
- 1 executar o código (submeter seu código para o interpretador Python);
- 2 chamar a função através do shell, fornecendo valor(es) de entrada;
- 3 analisar se o valor retornado é coerente, dadas as entradas e o que a função se propõe a calcular.

IDLE: executando uma função



Passo 1: Execute seu código a partir do editor do IDLE:
selecione **Run Module (F5)**

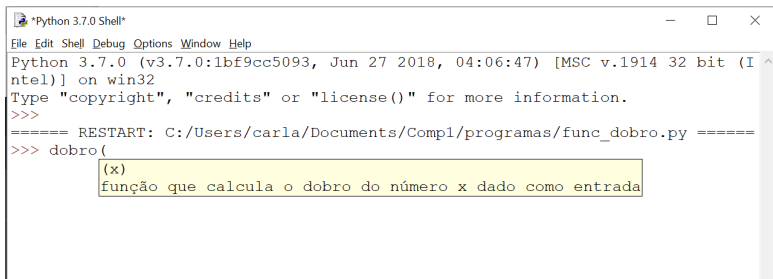
IDLE: executando uma função - passando do editor para o shell



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/carla/Documents/Comp1/programas/func_dobro.py =====
>>>
```

Ao mandar executar, seu código é submetido para o interpretador, e a janela do shell fica ativa (a janela do editor passa para trás)

IDLE: chamando uma função no shell

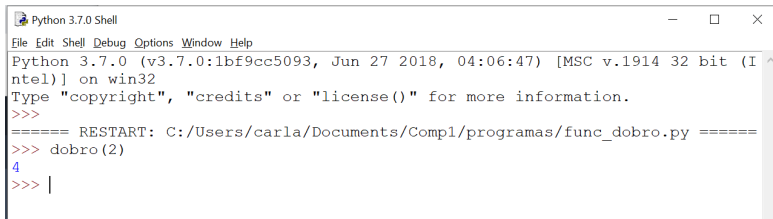


```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:/Users/carla/Documents/Comp1/programas/func_dobro.py =====
>>> dobro(
    (x)
    função que calcula o dobro do número x dado como entrada
```

Passo 2: Chamar a função através do shell, fornecendo valor(es) de entrada

Uma função previamente definida é "chamada" quando digitamos seu nome e botamos entre parênteses dados para cada uma de suas entradas.

IDLE: testando uma função no shell



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/carla/Documents/Comp1/programas/func_dobro.py =====
>>> dobro(2)
4
>>> |
```

Passo 3: analisar se o valor retornado é coerente, dadas as entradas e o que a função se propõe a calcular.

Computação I - Python

Aula 1 - Introdução à programação

Introdução ao uso de funções

Apresentado por: Carla A. D. M. Delgado

Produção DCC-UFRJ

Metodologia de referência <https://doi.org/10.5753/wei.2016.9683>

