UNIFCV
CENTRO UNIVERSITÁRIO CIDADE VERDE

Funções

Variáveis e parâmetros

Prof. Me. Leonardo G. Catharin

prof_leonardo@unifcv.edu.br

Forma geral e parâmetros [*arquivo: exemplos-referencia*]

- **Forma geral**

```
def <nome_funcao> (<parametros>):
    <corpo da função>
    return <retorno>
```

- Os objetos em Python são **sempre** passados por referência, enquanto que os tipos primitivos são passados como valores.

- Se o objeto passado por referência é **imutável**, então a função não conseguirá alterar o objeto.

- É possível definir um valor padrão para cada parâmetro.

```
def calcula_juros(valor, taxa = 10):
    . . .
```

Parâmetros

- Se o último parâmetro da função começa com (*), então todos os valores passados a partir daquele parâmetro são colocados em uma tupla.

```
def  funcao(x, *y): #y será considerado como uma tupla
        . . .
        return <retorno>
```

- Se o último parâmetro da função começa com (**), então todos os valores passados a partir daquele parâmetro são colocados em um dicionário (**é necessário passar a chave**).

```
def  funcao(x, y, **z): #z será considerado como um dicionário
        . . .
        return <retorno>

foma(1, 2, x = 3)
```

Variáveis e escopos

- Toda variável utilizada dentro de uma função tem escopo local.

```
def calcula_juros(valor, taxa):
    aux = valor * taxa #a variável aux só é acessível dentro do escopo da função calcula_juros
    . . .
```

- Variáveis globais.
  - Variável acessível em todas as funções do módulo.
  - Utilização da palavra reservada **global**.

- Em Python...

```
def soma(x, y):
    global total
    total = x + y
    print(total)

#programa principal
global total
total = 10
soma(3, 5)
print(total)
```

Retornos

- Utiliza-se a palavra reservada **return**.
- Caso nenhum retorno seja explicitado, a função retorna **None** (sem valor).
- **Curiosidade:** é possível que uma função tenha dois **retornos**?

Retornos

- Utiliza-se a palavra reservada **return**.
- Caso nenhum retorno seja explicitado, a função retorna **None** (sem valor).
- Curiosidade: é possível que uma função tenha dois **retornos** ?
  - Não, mas ela pode retornar dois valores ao mesmo tempo [*arquivo: exemplos-referencia*].
- O **return** tem duas funções:
  - Estabelecer o valor obtido a partir de uma expressão e;
  - Mudar o fluxo de execução do código.

Variável nativa __name__

- Todo arquivo Python é considerado um **módulo** no projeto.
- Todo módulo pode ser executado diretamente ou por **outro módulo.**
- Para executar um módulo diretamente, utiliza-se a variável **__name__**

```
If __name__ == '__main__':
    #código executado diretamente
```

- **__main__** é o nome atribuído ao módulo executado diretamente (por exemplo, via prompt).

Referências

- Ramalho, L. Python Fluente: Programação clara, concisa e eficaz. Novatec Editora, 2015.
- PET - ADS. Introdução à Programação com Python. IFSP – São Carlos.