



PYTHON PARA PLN

Introdução ao Python

Roney Lira de Sales Santos roneysantos@usp.br

Prof. Thiago A. S. Pardo

INTRODUÇÃO À LINGUAGEM

- Linguagem de **alto nível**
 - Ao ler o comando, já se presume o que ele significa!
- Vários tipos de programação:
 - Modular: **divisão** do algoritmo em blocos
 - Orientada a objetos: **classes** e **objetos** referenciados
 - Funcional: aplicação de **funções matemáticas**
- Tipagem **forte** e dinâmica
- Grande coleção de bibliotecas
- Código aberto

INTRODUÇÃO À LINGUAGEM

- Python é uma linguagem **interpretada**
- Não existe **etapa de compilação** de código
 - Assim como em C, C++, Java
- Basta **executar** o comando Python e pronto!

```
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 22:22:05) [MSC v.1916 64 bit  
(AMD64)] on win32  
Type "help", "copyright", "credits" or "license()" for more information.  
>>> print("Hello World")  
Hello World  
>>> print(1+1)  
2  
>>> print("roney" + "lira")  
roneylira  
>>> x = 4+6  
>>> x  
10  
>>> |
```

INTRODUÇÃO À LINGUAGEM

- Não é preciso terminar comandos com ;
- Não é preciso **declarar o tipo de dados** das variáveis

```
>>> x = 4+6
>>> x
10
>>> y = "roney lira"
>>> y
'roney lira'
>>> k = 1.8 - 1.5
>>> k
0.30000000000000004
>>> type(x)
<class 'int'>
>>> type(y)
<class 'str'>
>>> type(k)
<class 'float'>
>>> |
```

INTRODUÇÃO À LINGUAGEM

- Instalação e uso: site oficial do Python
 - <https://www.python.org/downloads/>



- Os códigos podem ser executados:
 - Em programas como **PyCharm** ou **Sublime**
 - por meio de linha de comando (prompt Windows, terminal Linux, IDLE Python...)
 - `python programa.py`
 - `./programa.py`

TIPOS DE DADOS

```
>>> x
10
>>> y = "roney lira"
>>> y
'roney lira'
>>> k = 1.8 - 1.5
>>> k
0.30000000000000004
>>> type(x)
<class 'int'>
>>> type(y)
<class 'str'>
>>> type(k)
<class 'float'>
>>> x > 6
True
>>> x > 15
False
>>> |
```

- Tipos de dados básicos
 - `int`, `string`, `float`, `bool`...
- O **tipo** de uma variável **muda** conforme o valor que lhe é atribuído.
 - Princípio da dinâmica
- **`type(var)`**

TIPAGEM FORTE

```
>>> v = "1"
>>> q = 1
>>> v + q
Traceback (most recent call last):
  File "<pyshell#16>", line 1, in <module>
    v + q
TypeError: can only concatenate str (not "int") to str
>>> |
```

- Ou seja, Python é uma linguagem que tem a característica de que não permite um mesmo dado ser tratado como se fosse de outro tipo.
- Mais sobre isso [aqui](#).

INDENTAÇÃO

- Bom, Python não usa nenhum limitante para estrutura de bloco, comum em outras linguagens
 - {, }, ;, **begin**, **end**, e por aí vai.
- A estrutura do bloco é definida pela indentação, ou seja, o alinhamento dos comandos é que define a estrutura!

```
if num % 2 == 0:  
    par = par + 1  
    print("PAR")  
else:  
    impar = impar + 1  
    print("IMPAR")
```

```
14 if num % 2 == 0:  
15 print("PAR")
```

```
File "C:\Users\roney\OneDrive\Documentos\10.  
PLN\aula8.py", line 15  
    print("PAR")  
    ^
```

```
IndentationError: expected an indented block  
[Finished in 0.4s]
```


PARA NOSSAS PRÁTICAS, USAREMOS...

- Listas e Tuplas
- Dicionários
- Arquivos
- Strings
 - Textos, cara do nosso PLN né! =)
- Claro, existem beeeem mais coisas relacionadas ao Python!
 - Estruturas de controle, funções, classes, programação funcional, uso de *frameworks* diversos...
- E também *frameworks* relacionados ao PLN, como **NLTK** e **spaCy**
 - **Em breve** nas telinhas do curso!

LISTAS E TUPLAS

- Estruturas de dados **nativas** da linguagem
 - **list** e **tuple**
- Informações dentro das listas e tuplas podem ser de tipos diferentes
- Acesso sequencial: fatias (*slicing*) ou diretamente
- Métodos prontos para adicionar, remover, ordenar, procurar, contar, entre vários outros
- Listas: **mutáveis** e delimitadas por **colchetes**
- Tuplas: **imutáveis** e delimitadas por **parênteses**

LISTAS - MÉTODOS

```
l = list(range(5))  
print(l)  
l.append(5)  
print(l)  
l.insert(0,6)  
print(l)  
l.reverse()  
print(l)  
l.sort()  
print(l)
```

```
[0, 1, 2, 3, 4]  
[0, 1, 2, 3, 4, 5]  
[6, 0, 1, 2, 3, 4, 5]  
[5, 4, 3, 2, 1, 0, 6]  
[0, 1, 2, 3, 4, 5, 6]
```

- **extend(L)**: inclusão de uma lista **l2** (**append**)
- **remove(x)**: remove a primeira ocorrência de **x**
- **index(x)**: índice da primeira ocorrência de **x**
- **count(x)**: número de ocorrências de **x** na lista