

APRENDIZAGEM DE MÁQUINA

(usando Python)

Thiago Marzagão

INTRODUÇÃO À PROGRAMAÇÃO



Guido van Rossum (BDFL - Benevolent Dictator For Life)



Guido van Rossum (BDFL - Benevolent Dictator For Life)



por que Python?

- open source
- excelente p/ mineração de dados
- amplamente usado (ou seja, é fácil conseguir ajuda)
- facilita “appificar” seu modelo
- boa linguagem p/ iniciantes: simples e intuitiva

o fundamental

- tipos de dados (str, int, float, etc)
- variáveis
- condicionais (if/else)
- operadores (and/or/not)
- loops
- listas, conjuntos, dicionários
- REPL vs scripts
- I/O

tipos de dados

- $1 + 1$
- "1" + "1"
- "oi" + "mundo"
- $1.5 + 1.5$
- "1.5" + "1.5"
- $4 / 2$
- $5 / 2$
- $5.0 / 2$
- $1 + 1.5$
- $1 + \text{"oi"}$
- "1" + "oi"
- `type(5)`, `type("5")`, `type(5.0)`

variáveis

- $x = 3$
- $y = 4$
- $x + y$
- $x - y$
- $x * y$
- x / y
- $\text{float}(x) / y$
- $x ** y$
- $((x + y) * (x - y) + (x ** y)) + 1000.5$
- $z = ((x + y) * (x - y) + (x ** y)) + 1000.5$

variáveis (cont.)

- $x = 3$
- $y = 4$
- $x + y$
- $x = 1$
- $x + y$
- variáveis variam!
- $z = x + y$
- $x = 10$
- z
- mas nem sempre como se espera (mais sobre isso ao longo do curso)

listas

- `x = [1, 3, 5, 7, 9]`
- `type(x)`
- `len(x)`
- `x.append(11)`
- `x`
- `x.remove(1)`
- `x`
- `y = [2, 4, 6, 8, 10]`
- `z = x + y`
- `z`
- `z[0]` (Python conta a partir de zero.)
- `z[3] = 20`
- `sorted(z)`
- `z[-1]`

listas (cont.)

- listas podem diferentes tipos de dados
- `x = [1.1, 201.312, 3123.8]`
- `x = ["joao", "maria", "priscila", "alexandre"]`
- `x = ["joao", 1, 5.5]`
- listas podem contar outras listas
- `x = [[1, 2, 3], [4, 5, 6]]`
- `type(x)`
- `len(x)`
- `x[0]`
- `x = [[[1, 2, 3], [4, 5, 6]], [[7, 8, 9], [10, 11, 12]]]`
- `len(x)`
- `x[0]`
- `x[0][0]`
- `x[0][0][0]`
- `x[0][0][0] = 15`

conjuntos (sets)

- parecidos c/ listas
- mas sem elementos repetidos
- e nao é possível ordenar os elementos
- $x = [1, 3, 5, 5, 7, 9, 9]$
- $s = \text{set}(x)$
- s
- $\text{type}(s)$
- $\text{len}(s)$
- uso freqüente: $\text{len}(\text{set}(x))$

dicionários

- são como “listas telefônicas”
- mapeiam um conjunto de elementos a outro
- exemplo: alunos -> notas
- $d = \{ \text{“joao”}: 5.5, \text{“maria”}: 7.2, \text{“priscila”}: 6.8 \}$
- `type(d)`
- `d[“joao”]`
- `d[“maria”]`
- `d[“priscila”]`
- `d[“joao”] = 6.5`
- `d[“alexandre”] = 7.1`
- `d`
- `d.keys()`
- `d.values()`

FOR loops

- `x = [1, 3, 5, 7, 9]`
- `for i in x:`
- `print(i ** 2)`
- (importante: 4 espaços de “margem”)

condicionais

- `x = [1, 3, 5, 7, 9]`
- `for i in x:`
- `print(i ** 2)`
- (importante: 4 espaços de “margem”)
- `for i in x:`
- `if i > 5:`
- `print(i ** 2)`
- (importante: mais 4 espaços de “margem”)
- (4 espaços de margem depois de “for” e depois de “if”)

operadores

- $1 < 2$
- $2 < 1$
- $(1 < 2)$ or $(2 < 1)$
- $(1 < 2)$ and $(2 < 1)$
- $(1 < 2)$ and not $(2 < 1)$
- $((1 < 2)$ and $(2 < 1))$ or $((1 < 2)$ or $(2 < 1))$
- $z = ((x + y) * (x - y) + (x ** y)) + 1000.5$
- if $z < 1200$:
 - `print("z menor que 1200")`
- else:
 - `print("z maior ou igual a 1200")`

- `def potencias(numero):`
- `quadrado = numero ** 2`
- `cubo = quadrado ** 3`
- `quarta = cubo ** 4`
- `quinta = quarta ** 2`
- `return quinta`
- (4 espaços de margem depois de “for”, depois de “if” e depois de “def”)
- `potencias(2)`
- `potencias(3)`
- `x = potencias(2)`
- `y = potencias(3)`

funções (cont.)

- def macarena(nome):
- texto = ""
- texto = texto + nome + "!"
- texto += " baila"
- texto += " tu cuerpo"
- texto += " alegria"
- texto += " macarena"
- texto += " hey macarena!"
- return texto
- (4 espaços de margem depois de "for", depois de "if" e depois de "def")
- macarena("priscila")

funções (cont.)

- `def potencia2(numero1, numero2):`
- `numero = numero1 ** numero2`
- `return numero`
- (4 espaços de margem depois de “for”, depois de “if” e depois de “def”)
- `potencia2(5, 4)`

funções (cont.)

- por que funções?
- DRY = Don't Repeat Yourself

pacotes

- pacotes são conjuntos de funções
- `import math`
- `math.sqrt(4)`
- `import random`
- `random.random()`
- alguns pacotes já vêm no Python mas a maioria precisa ser baixada e instalada
- pacotes que vamos usar (muito!) no curso: `pandas`; `scikit-learn`

onde conseguir ajuda

- Google
- StackOverflow.com
- pt.StackOverflow.com

Cutting corners to meet arbitrary management deadlines



Essential

Copying and Pasting from Stack Overflow

O'REILLY®

The Practical Developer
@ThePracticalDev

Software can be chaotic, but we make it work



Expert

Trying Stuff Until it Works

ORLY?

*The Practical Developer
@ThePracticalDev*