Programação Python

Aula 01: Muito Prazer, Linguagem Python

Prof. Eduardo Corrêa Gonçalves

22/03/2021

Sumário

Introdução (1/5)

- **O que é Python?**
  - Linguagem de programação de **propósito geral**.
    - Serve para:
      - Ciência de Dados*;
      - Aplicações Web;
      - Scripts de Automação;
      - *E outras aplicações*...

- **FILOSOFIA**:

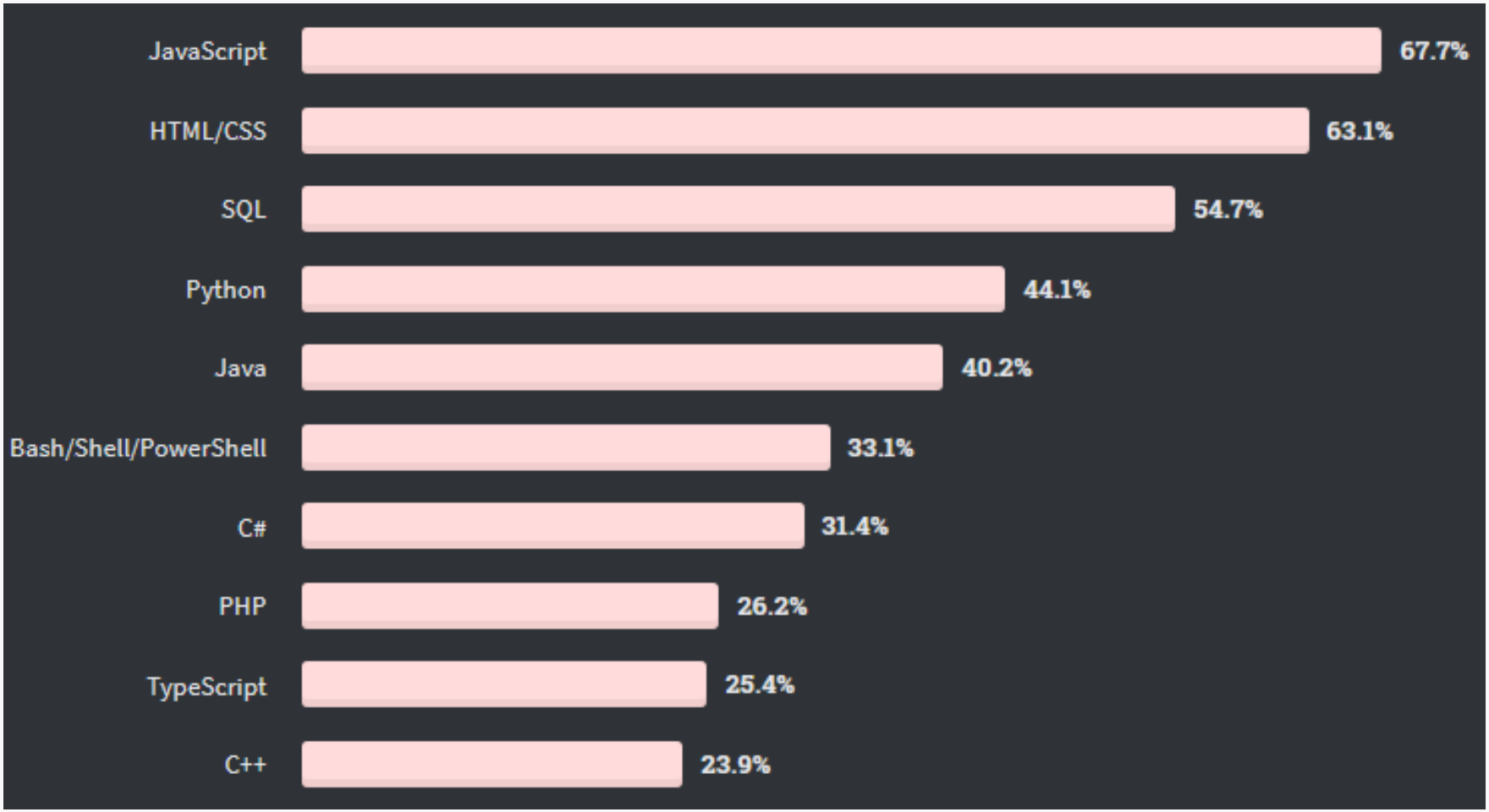
  *Priorizar a construção de programas **simples** e **legíveis** (“bonitos”) sobre a de programas que, mesmo eficientes, sejam complicados e pouco legíveis (“feios”)”*

- A **versão** atual da linguagem é **3.9**

* Ao longo deste curso, usarei os termos “Estatística” e “Ciência de Dados” de forma intercambiável.

Introdução (2/5)

- **É uma linguagem muito popular***



* Segundo os resultados da **Stack Overflow 2020 Developer Survey**: https://insights.stackoverflow.com/survey/2020

Introdução (3/5)

- **É a mais popular para Ciência de Dados.**
  - Abaixo o resultado da análise dos requisitos exigidos em 1.170 ofertas de emprego nos EUA*

- Fonte: https://www.kdnuggets.com/2020/08/data-scientist-job-market-2020.html

Introdução (4/5)

- **Por que a linguagem Python é muito legal para Estatística ?**
  - Ela é **interpretada** e pode ser usada de forma **interativa**.
    - No modo interativo, cada comando digitado pode ser imediatamente traduzido e executado.
    - Com isso, resultados intermediários de um processo de análise estatística podem ser examinados em tempo real.
  - É **extensível** através de **pacotes** (*há dezenas de milhares*)
    - NumPy, pandas, scikit-learn, SciPy, Matplotlib, NLTK, Keras, ...
  - É **gratuita**.
  - É **multiparadigma**.
    - Programação procedural (“tradicional”).
    - Programação orientada a objetos.
    - Programação funcional.

Introdução (5/5)

- **Mas Python é bem versátil ... Não serve só para Estatística!**
  - A tabela abaixo apresenta as principais aplicações do Python*

Pos.	Aplicação	%
1.	**Data analysis**	55%
2.	Web development	50%
3.	**Machine learning**	40%
4.	DevOps / System Adm / Writing automation scripts	38%
5.	**Programming of web parsers / scrapers / crawlers**	36%
6.	Software testing / Writing automated tests	29%
7.	**Educational purposes**	27%
8.	Software prototyping	23%
	...	...
12.	Game development	9%
13.	Embedded development	8%
14.	Mobile development	7%

* Segundo os resultados da **Python Developers Survey 2020**: https://www.jetbrains.com/lp/python-developers-survey-2020/

Python – Escolhendo um Ambiente (0/5)

- **PASSO 0: Baixar o livrinho**
  - Se você não sabe nada de Python, pode começar baixando o livro gratuito e oficial do nosso curso:
    - **Meu Primeiro Livro de Python** (*livro para iniciantes*)
      - https://github.com/edubd/meu_primeiro_livro_de_python
  - Outros recursos interessantes são:
    - **Python 3 tutorial** (*tutorial de nível intermediário*)
      - https://www.python-course.eu/python3_course.php
    - Artigos do site **RealPython** (*eu considero o melhor site sobre Python*)
    - Artigos de outros sites, como **DataCamp**, **KDNuggets**, **Towards Data Science** etc. (*sites específicos de Ciência de Dados*)

Python – Escolhendo um Ambiente (1/5)

- **PASSO 1: Escolher um Ambiente / Distribuição**
  - Python possui um impressionante ecossistema com dezenas de milhares de pacotes!
    - Isso motivou o surgimento de diversas **distribuições** do ambiente Python.
    - Uma distribuição é um arquivo instalador que reúne:

interpretador Python + standard library (biblioteca padrão) + coleção de pacotes selecionados.

  - Cada distribuição é voltada para um propósito específico.
    - Algumas são para Ciência de Dados, outras para desenvolvimento web etc.
    - Por isso, cada distribuição vem com pacotes específicos.

Python – Escolhendo um Ambiente (2/5)

- **Neste curso você pode usar as seguintes distribuições:**

- **CPython:** é a distribuição Python "neutra e oficial", obtida em www.python.org.
  - É preciso instalar os pacotes para ciência de dados separadamente.
  - Instruções no "Meu Primeiro Livro de Python", Anexo A.

- **WinPython**: distribuição voltada para Ciência de Dados
  - Boa para iniciantes, mas ocupa 3,5GB em disco.
  - Instruções no "Meu Primeiro Livro de Python", pag 15-22.

- **Thonny**: para propósitos educacionais, leve, serve apenas para as aulas iniciais.
  - Você pode baixar no meu google drive, descompactar e já estará pronto pra uso:
  - https://drive.google.com/drive/folders/1jEvAOeXnx8N8F1D1ziRFEDwz0B2WrcE1

- **Obs1***: é possível instalar o CPython, WinPython e Thonny na mesma máquina (um não interferirá no outro)*
- **Obs2***: a distribuição **Anaconda** é a mais popular para Ciência de Dados. Se você é um pouco mais experiente, pode utilizá-la.*

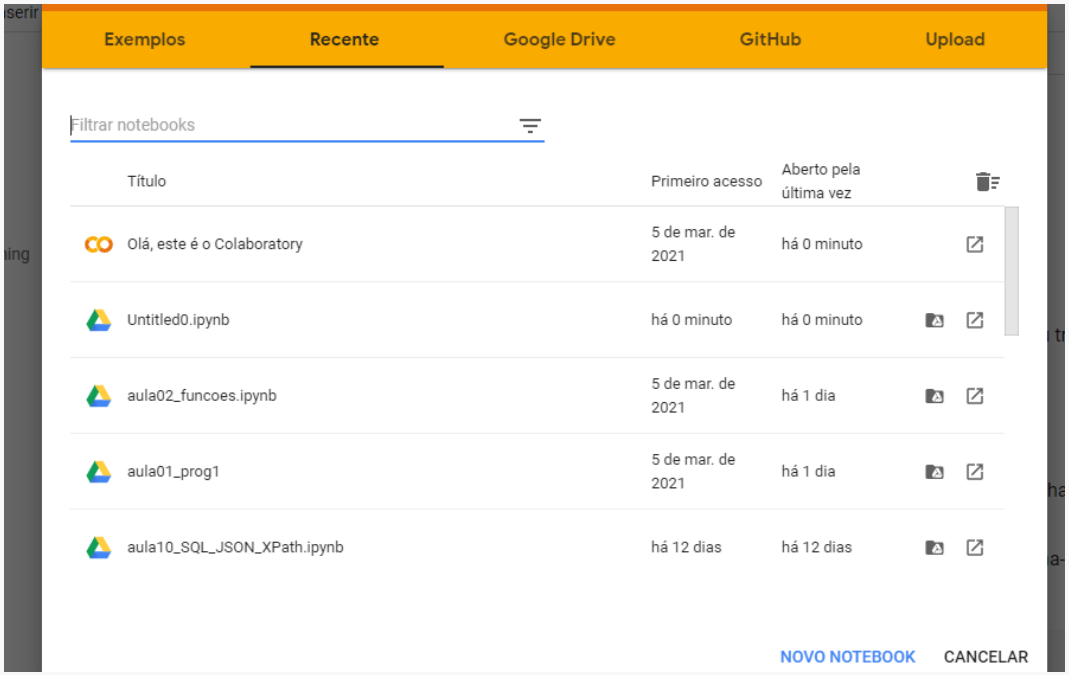
Python – Escolhendo um Ambiente (3/5)

- **... Ou você pode utilizar o Google Colab**
  - Não requer instalação em sua máquina basta ter uma conta Google!
  - Você escreve o código em seu navegador, gravando seus projetos (chamados “notebooks”) na nuvem.
    - Assim, pode acessá-los de qualquer lugar..
  - Ambiente voltado para Ciência de Dados (principais pacotes estão disponíveis).

Python – Escolhendo um Ambiente (4/5)

- **Para começar a trabalhar com Google Colab siga estes passos:**

  1. Efetue o login em sua conta do Google.
  2. Acesse o site https://colab.research.google.com
  3. Clique em “Novo Notebook”



- Mais informações em:
  - https://colab.research.google.com/notebooks/intro.ipynb
  - https://www.alura.com.br/artigos/google-colab-o-que-e-e-como-usar

Python – Escolhendo um Ambiente (5/5)

- **Também deverá escolher um Ambiente de Desenvolvimento (IDE)**
  - **IDE** (*Integrated Development Environment*) é o software que usamos para criar, testar e executar os programas Python.
    - **Colab e Thonny**: a IDE já está embutida no ambiente.
    - **CPython**: vem apenas com a python shell, mas você pode instalar a Spyder.
      - pag. 19-21 e Anexo A do “Meu Primeiro Livro de Python”
    - **WinPython**: vem com o IDLE (python shell), Jupyter e Spyder
      - A Spyder é apresentada nas pags. 21-23 do “Meu Primeiro Livro de Python”

Programação – Primeiros Passos (1/10)

- **Programa 1: Variáveis e tipos; funções type() e print()**

```
#PARTE 1: declaração de variáveis
nome = 'Jane'
sobrenome = "Austen"
idade = 41; nota = 9.9; aprovado = True

#PARTE 2: imprime o conteúdo das variáveis e seus tipos
print(nome, sobrenome, idade, nota, aprovado)
print(type(nome), type(sobrenome), type(idade),
type(nota), type(aprovado))

#PARTE 3: mudando o valor e o tipo da variável "nota"
nota = 'A'
print('mudei o valor e o tipo de "nota" para: ',
nota, ",", type(nota))
```

```
>>>
Jane Austen 41 9.9 True
<class 'str'> <class 'str'> <class 'int'> <class 'float'> <class 'bool'>
mudei o valor e o tipo de "nota" para:  A , <class 'str'>
```

Programação – Primeiros Passos (2/10)

- **Observações**
  - Há **diferenciação** entre letras **maiúsculas e minúsculas** nos nomes de comandos, funções e variáveis.
    - Se você nomear uma variável como **x**, não poderá referenciá-la como **X**.
  - Em geral recomenda-se usar nomes de variáveis descritivos, escritos em minúsculo, usando *underscore* como separador. Exemplos:
    - idade
    - renda_media_anual
  - Python possui um comando para entrada de dados via teclado chamado input().
    - Porém, você dificilmente o utilizará.
    - **Motivo**: na Ciência de Dados, quase sempre a entrada de dados é feita via arquivos.

Programação – Primeiros Passos (3/10)

- **Programa 2: Operadores Matemáticos**
  - Nos exemplos, considere x = 5 e y = 2.

Operação	Símbolo	Exemplo	Result.
Adição, Subtração, Multiplicação e Divisão	+, -, *, /	(x + y) * 2	14
Exponenciação	**	y**x	32
Quociente da Divisão Inteira	//	x // y	2
Resto da Divisão Inteira	%	x % y	1

- **Obs.:** a divisão de dois inteiros **sempre** gera um float

```
a=10; b=5; c=a/b
print(a, b, c)
print(type(a), type(b), type(c))
```

```
>>>
10 5 2.0
<class 'int'> <class 'int'> <class 'float'>
```

Programação – Primeiros Passos (4/10)

- **Desvio Condicional – if e else**
  - Instruções if, elif e else.
  - Linhas subordinadas devem ser **indentadas**.
    - Padrão = 4 espaços (Colab usa 2 espaços).

```
idade = 17
if (idade >= 18):
  print("Pode entrar, a festa está bombando!")
  print("Temos muita música e drinks especiais!!!")
else:
  print("Você é jovem demais para este clube!")
  print("Volte apenas quando fizer 18.")
```

- Se você **não indentar**, vai receber o **erro**:
  - IndentationError: expected an indented block

Programação – Primeiros Passos (5/10)

- **Operadores Relacionais**

x **==** y	O valor de x é igual ao de y?
x **!=** y	O valor de x é diferente do de y?
x **>** y	O valor de x é maior que o de y?
x **>=** y	O valor de x é maior ou igual ao de y?
x **<** y	O valor de x é menor que o de y?
x **<=** y	O valor de x é menor ou igual ao de y?
x **is** y	x e y apontam para o mesmo endereço de memória? (detalhes em uma próxima aula)

- **Operadores Lógicos**

and	A sentença é verdadeira se todas as condições forem verdadeiras.
or	A sentença é verdadeira se uma das condições for verdadeira
not	Inverte o valor lógico de uma sentença.

Programação – Primeiros Passos (6/10)

- **Desvio Condicional – elif**
  - **elif** faz o papel de “else if” (senão se).

```
idade = 25

if (idade < 18):
  faixa_etaria = '<18'
elif (idade >= 18 and idade < 30):
  faixa_etaria = '18-29'
elif (idade >= 30 and idade < 40):
  faixa_etaria = '30-39'
else:
  faixa_etaria = '>=40'

print('Se a idade é', idade, 'então a faixa etária é :',
      faixa_etaria)
```

```
>>> Se a idade é 25 então a faixa etária é : 18-29
```

Programação – Primeiros Passos (7/10)

- **Repetição com <u>while</u>**
  - Enquanto condição for VERDADEIRA, bloco de código é executado.

```
c = -20
print('Tabela de conversão')
print('graus Celsius x Fahrenheit')

while c <= 100:
  f = c*1.8 + 32
  print(c,'°C ----> ',f,'°F')
  c += 10

print('FIM!!!')
```

```
>>>
Tabela de conversão
graus Celsius x Fahrenheit
-20 ºC ---->  -4.0 ºF
-10 ºC ---->  14.0 ºF
0 ºC ---->  32.0 ºF
10 ºC ---->  50.0 ºF
20 ºC ---->  68.0 ºF
30 ºC ---->  86.0 ºF
40 ºC ---->  104.0 ºF
50 ºC ---->  122.0 ºF
60 ºC ---->  140.0 ºF
70 ºC ---->  158.0 ºF
80 ºC ---->  176.0 ºF
90 ºC ---->  194.0 ºF
100 ºC ---->  212.0 ºF
FIM!!!
```

Programação – Primeiros Passos (8/10)

- **Repetição com for-range()**
  - Na linguagem Python o **for** pode iterar apenas sobre **sequências**.
  - Para implementar um for básico, usamos a função **range()** para gerar as sequências.

- **Sintaxe**: **range**(início, fim, incremento)
  - **início**: número inicial da sequência (opcional). Caso seja omitido, o valor 0 é assumido.
  - **fim**: a sequência será gerada até, mas sem incluir, o número especificado neste parâmetro (único parâmetro obrigatório).
  - **incremento**: diferença entre cada número na sequência (opcional). Se omitido, o valor 1 é adotado.

```
range(3)          #[0,1,2]
range(1,4)        #[1,2,3]
range(0,10,2)     #[0,2,4,6,8]
```

Programação – Primeiros Passos (9/10)

- **Repetição com for-range()**

```
print('\n* * imprimindo de 0 a 3')
for i in range(4): print(i)

print('\n* * imprimindo de 10 a 15')
for i in range(10, 16):
  print(i)

print('\n* * ordem reversa: 3, 2, 1')
for i in range(3, 0, -1):
  print(i)
```

```
>>>
* * imprimindo de 0 a 3
0
1
2
3

* * imprimindo de 10 a 15
10
11
12
13
14
15

* * ordem reversa: 3, 2, 1
3
2
1
```

Programação – Primeiros Passos (10/10)

- **Observações**
  - Os comandos **break** e **continue** também existem no Python.
    - **break** serve para quebrar um laço “na marra” (*muito útil !!!*)
    - **continue** serve para quebrar uma iteração.
  - Os principais pacotes para Ciência de Dados permitem a execução de muitas operações sobre conjuntos de dados **sem a necessidade da implementação de laços**.
    - Esse processo é conhecido como computação vetorizada (*vectorization*).
    - Ele será apresentado em detalhes a partir da aula sobre a biblioteca NumPy.
    - Por esta razão, o comando while é menos empregado na Ciência de Dados em comparação ao seu uso em código de aplicativos.

Tarefas

- Escolha um ambiente Python (CPython, WinPython, Thonny ou Colab) e uma IDE. Em seguida faça os exercícios a seguir:

(1) – Sendo “p”, “q” e “r” variáveis cujos conteúdos são iguais a 2, 3 e 12, respectivamente, faça um programa que calcule e imprima:

$a = 100 \times \text{QUOCIENTE}(q, p) + r$
$b = p \times \text{RESTO}(r,5) - q \div 2$

(2) – Faça um programa que troque os valores de 3 variáveis “a”, “b” e “c”, de modo que “a” contenha o menor valor, “b” o valor intermediário e “c” o maior valor.
Ex.: se a=10, b=1 e c=5, o programa deve trocar para a=1, b=5 e c=10.

(3) – Faça um programa imprima os resultados das divisões sucessivas de um número real *x* por 2 enquanto o valor resultante da divisão for maior do que 0,01

(4) – Usando for-range(), faça um programa que calcule e imprima o valor de “s”:

$s = (1/1) + (3/2) + (5/3) + (7/4) + \ldots + (99/50)$

Referências

- Corrêa, E. (2020). “Meu Primeiro Livro de Python”. V 2.0.0, edubd, 2020. (*capítulo 1 e Anexo A*).
  - Disponível em: https://github.com/edubd/meu_primeiro_livro_de_python