Minicurso de Python para Matemática

Pedro H A Konzen

3 de setembro de 2021

Sumário

Licença

1 Sobre a linguagem

Python é uma linguagem de programação de alto nível e multi-paradigma. Ou seja, é relativamente próxima das linguagens humanas naturais, é desenvolvida para aplicações diversas e permite a utilização de diferentes paradigmas de programação (programação estruturada, orientada a objetos, orientada a eventos, paralelização, etc.).

- Site oficial: https://www.python.org/

1.1 Instalação e execução

Para executar um código Python é necessário instalar um interpretador. No site oficial do Python estão disponível para *download* interpretadores gratuitos e com licença para uso livre. Neste minicurso, vamos utilizar Python 3 instalado em um sistema `Linux`. Para outros sistemas, pode ser necessário fazer algumas pequenas adequações.

1.2 Utilização

A execução de códigos Pythonpode ser feita de três formas básicas:

- em modo iterativa em um console Python;
- por execução de um código `.py` em um console Python;
- por execução de um cógido `.py` em um terminal;

Exemplo 1.1. Implemente o seguinte pseudocódigo.

```
s = "Ola, mundo!".
imprime(s). (imprime a string s)
```

a) em modo iterativo no console;

b) escrevendo o código `.py` e executando-o no console;

c) escrevendo o código `.py` e executando-o no terminal.

Resolução. Seguem as implementações em cada caso.

a) Em modo iterativo.

Iniciamos um console Python em terminal digitando

```
$ python3
```

Então, digitamos

```
>>> s = "Ola, Mundo!"                    1
>>> print(s) #imprime a string s         2
Ola, Mundo!                              3
```

Para encerrar o console, digitamos

```
>>> quit()                               1
```

b) Executando *script* `.py` no console.

Primeiramente, escrevemos o código

```
s = "Ola, Mundo!"                        1
print(s) # imprime a string s            2
```

em um editor de texto (ou no seu IDE de preferência) e salvamo-lo em `/pasta/codigo.py`. Então, executamo-lo no console Python com

```
>>> exec(open('/pasta/codigo.py').read())   1
Ola, mundo!                                 2
```

c) Executando o código em terminal.

Considerando que já temos o código salvo em `/pasta/codigo.py`, executamo-lo com

```
$ python3 /pasta/codigo.py
Olá, mundo!
```

2 Elementos da linguagem

Python é uma linguagem de programação dinâmica em que as variáveis são declaradas automaticamente ao receberem um valor. Por exemplo, consideremos as seguintes instruções

```
>>> x = 1                                1
>>> y = x * 2.0                          2
```

Na primeira instrução, a variável `x` é recebe o valor inteiro 1 e, então, é armazenado na memória do computador como um objeto da classe `int`. Na segunda instrução, `y` recebe o valor decimal 2.0 (resultado de 1×2.0) e é armazenado como um objeto da classe `float` (ponto flutuante de 64-*bits*). Podemos verificar isso, com as seguintes instruções

```
>>> print(x, y)                                    1
1 2.0                                              2
>>> print(type(x), type(y))                        3
<class 'int'> <class 'float'>                      4
```

Códigos Python admitem comentários e continuação de linha como no seguinte exemplo

```
>>> # isso eh um comentario                       1
>>> s = "isso eh uma \                             2
... string"                                        3
>>> print(s)                                       4
isso eh uma string                                 5
>>> type(s)                                        6
<class 'str'>                                      7
```

Observação 2.1. (Notação científica) O Python aceita notação científca, por exemplo 5.2×10^{-2} é digitado como

```
>>> 5.2e-2                                         1
0.052                                              2
```

Observação 2.2. Além dos tipos numéricos e *string*, Python também conta com os tipos de dados `list` (lista), `tuple` (n-upla) e `dict` (dicionário). Estudaremos estes tipos mais adiante neste minicurso.

2.1 Operações aritméticas elementares

Os operadores aritméticos elementares são:

`+`: adição

`-`: subtração

`*`: multiplicação

`/`: divisão

`**`: potenciação

`%`: módulo

`//`: módulo

Consideremos o seguinte exemplo

```
>>> 2+8*3/2**2-1
7.0
```

Observamos que as operações `**` tem precedência sobre as operações `*, /`, as quais têm precedência sobre as operações `+, -`. Operações de mesma precedência seguem a ordem da esquerda para direita, conforme escritas na linha de comando. Usa-se parênteses para alterar a precedência entre as operações, por exemplo

```
>>> (2+8*3)/2**2-1
5.5
```

Consulte mais informações sobre a precedência de operadores em Python Docs.

Exercício 2.1. Compute as raízes do seguinte polinômio quadrático

$$x^2 - x - 2 \tag{1}$$

usando a fórmula de Bhaskara[1]

O operador `%` módulo computa o resto da divisão e o operador `//` a divisão inteira, por exemplo

```
>>> 5 % 2
1
>>> 5 // 2
2
```

Exercício 2.2. Use o Python para verificar se 14/21 é menor ou igual a 15/23. Então, compute o resto da divisão do maior quociente.

[1]Bhaskara Akaria, 1114 - 1185, matemático e astrônomo indiano. Fonte: Wikipédia.

2.2 Funções e constantes elementares

O módulo Python math disponibiliza várias funções e constantes elementares. Para usá-las, precisamos importar o módulo para nossa seção. Fazemos isso com a instrução

```
>>> import math
```

Com isso, temos acesso a todas as definições e declarações contidas neste módulo. Por exemplo

```
>>> math.pi
3.141592653589793
>>> math.cos(math.pi)
-1.0
>>> math.sqrt(2)
1.4142135623730951
>>> math.log(math.e)
1.0
```

Observação 2.3. Notemos que `math.log` é a função logaritmo natural, i.e. $\ln(x) = \log_e(x)$. A implementação Python para o logaritmo de base 10 é `math.log(x,10)` ou, mais acurado, `math.log10`.

Exercício 2.3. Compute $e^{\log_3(\pi)}$.

2.3 Operadores de comparação elementares

Os operadores de comparação elementares são

`==`: igual a

`!=`: diferente de

`>`: maior que

`<`: menor que

`>=`: maior ou igual que

`<=`: menor ou igual que

Estes operadores retornam os valores lógicos `True` (verdadeiro) ou `False` (falso).

Por exemplo, temos

```
>>> x = 2
>>> x + x == 4
True
```

Exercício 2.4. Atribua a variável `x` o valor $\sqrt{3}$. Então, verifique se o valor computado de x^2 é maior que 3. Em caso negativo, verifique se x^2 é menor que 3. Comente o resultado obtido.

2.4 Operações lógicos elementares

Os operadores lógicos elementares são:

`and`: e lógico

`or`: ou lógico

`not`: não lógico

A tabela booleana[2] do "e" lógico é

Valor	Valor	Resultado
True	True	True
True	False	False
False	True	False
False	False	False

Podemos verificar isso no Python como segue

```
>>> True and True
True
>>> True and False
False
>>> False and True
False
>>> False and False
False
```

Exercício 2.5. Construa as tabelas booleanas do operador `or` e do `not`.

[2]George Boole, 1815 - 1864, matemático e filósofo britânico. Fonte: Wikipédia.

Exercício 2.6. Use Python para verificar se $1.4 <= \sqrt{2} < 1.5$. E, também, verifique se $\sqrt{3} > 1.7$ ou $\sqrt{3} >= 1.7321$.

Em construção ...

Referências

[1] **REAMAT - Cálculo Numérico - Um Livro Colaborativo - Versão Python**. https://www.ufrgs.br/reamat/CalculoNumerico/livro-py/main.html, 2021. Apêndice A - Rápida introdução ao Python.