Professor Rodrigo Bossini
https://sites.google.com/site/professorrodrigobossini/

Introdução à Linguagem Python (estrutura sequencial)

Introdução

Este material é uma introdução sobre a linguagem Python, versão 3.x.x. Faça o download e instalação a partir do Link 1.1.

Link 1.1
https://www.python.org/

Em particular, caso tenha interesse em bibliotecas específicas para Ciência de Dados, Mineração de Dados e afins, você pode optar por instalar a distribuição Anaconda (Link 1.2). Ela já contém uma versão de Python e diversas bibliotecas para esse fim.

Link 1.2
https://www.anaconda.com/

1 Conceitos fundamentais

1.1 Atualmente, a linguagem Python possui **duas versões ativas**: 2 e 3. A versão 2 tem seu EOL (End of Life, que significa o fim do suporte) definido para 2020 [1]. Por essa e outras razões, iremos utilizar a versão 3.

1.2 Objetos são a principal coisa que programas em Python manipulam. Cada objeto tem um **tipo** que define o que é possível fazer com o objeto.

1.3 Objetos podem ser **escalares ou não escalares**. Objetos escalares são **indivisíveis**. Os não escalres possuem **estrutura interna**.

1.4 A linguagem Python possui **quatro tipos de objetos escalares**

- **int**, para representar números inteiros
- **float**, para representar números reais
- **bool**, para representar valores booleanos
- **None**, é um tipo com valor único. Veremos seu uso ao longo do curso

1.5 A função **type** permite que se descubra o tipo de um determinado objeto. Por exemplo: **type(1)** verifica o tipo do objeto 1.

1.6 Os **operadores aritméticos** são exibidos na Tabela 1.1.

Tabela 1.1

Operador	Significado	Exemplo	Resultado
+	Soma	2 + 3	5
-	Subtração	2 – 3	-1
*	Multiplicação	2 * 3	6
/	Divisão real	3 / 2	1.5
//	Divisão inteira	3 / 2	1
%	Resto da divisão	3 % 2	1
**	Potenciação	3 ** 2	9

1.7 Os **operadores de comparação** são exibidos na Tabela 1.2.

Tabela 1.2

Operador	Significado	Exemplo	Resultado
==	Igualdade	1 == 1 2 == 1	True False
!=	Diferença	1 != 1 2 != 1	False True
>	Maior	1 > 1 2 > 1	False True
>=	Maior ou igual	1 >= 1 2 >= 1 1 >= 2	True True False
<	Menor	1 < 1 2 < 1 1 < 2	False False True
<=	Menor ou igual	1 <= 1 2 <= 1 1 <= 2	True False True

1.8 Os **operadores lógicos** são exibidos na Tabela 1.3.

Tabela 1.3

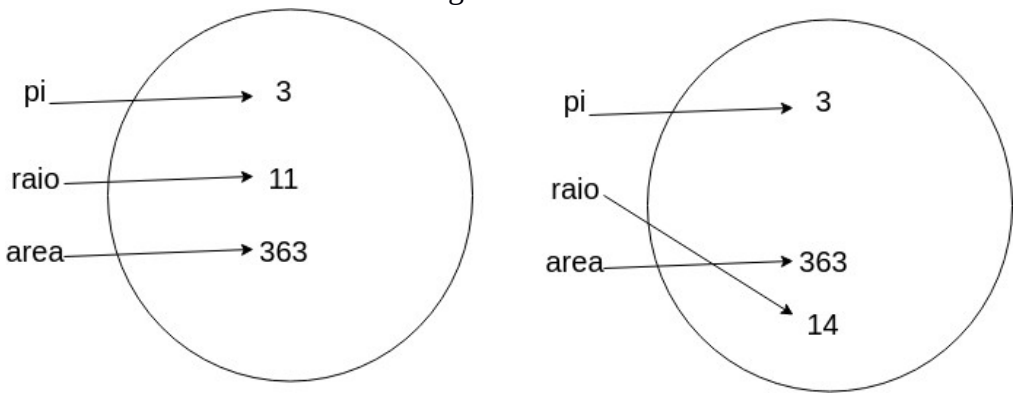
Operador	Significado	Exemplo	Resultado
and	“e” lógico	True and True True and False False and True False and False	True False False False
or	“ou” lógico	True or True True or False False or True False or False	True True True False
not	“não”	not True not False	False True

1.9 O **operador de atribuição** é o “=”. Veja um exemplo na Listagem 1.1, ilustrado na Figura 1.1.

Listagem 1.1

```
pi  =  3
raio = 11
area = pi * ( raio ** 2)
raio = 14
```

Figura 1.1



Também é possível fazer **atribuição múltipla**, como mostra a Listagem 1.2.

Listagem 1.2

```
x, y = 2, 3
y, x = x, y
```

1.10 Os **operadores compostos** permitem realizar operações aritméticas simultaneamente com uma atribução. Veja a Tabela 1.4.

Tabela 1.4

Operador	Significado	Exemplo	Resultado
+=	soma e atribuição	x = 1 x += 2	x = 3
-=	subtração e atribuição	x = 1 x -= 2	x = -1
*=	multiplicação e atribuição	x = 2 x *= 3	x = 6
/=	divisão real e atribuição	x = 5 x /= 2	x = 2.5
//=	divisão inteira e atribuição	x = 5 x //= 2	x = 2
%=	módulo e atribuição	x = 5 x %= 2	x = 1
**=	exponenciação e atribuição	x = 2 x **= 3	x = 8

1.11 A Tabela 1.5 mostra as **palavras reservadas** da linguagem Python (as versões 2 e 3 possuem algumas diferenças)

Tabela 1.5

False	None	True	and	as	assert
break	class	continue	def	del	elif
else	except	finally	for	from	global
if	import	in	is	lambda	nonlocal
not	or	pass	raise	return	try
while	with	yield			

Nota: Use o código da Listagem 1.3 para verificar a lista de palavras reservadas da versão de Python que você está usando.

Listagem 1.3

```
import keyword
keyword.kwlist
```

1.12 Os **nomes de variáveis** em Python devem estar de acordo com as seguintes regras:

- composição por letras maiúsculas, letras minúsculas, dígitos e o símbolo *underscore* (_).
- não podem começar com dígitos
- não podem ser iguais às palavras reservadas da linguagem

A Tabela 1.6 mostra exemplos de nomes de variáveis válidos e inválidos.

Tabela 1.6

Nomes válidos	Nomes inválidos
a	1
ba	1a
Baa	A$
_	1____
_C_	#aa
_____	a#
_____1	for

1.13 Os **comentários** em Python são feitos com o símbolo #.

1.14 Strings em Python são do tipo “str”. No que diz respeito a strings, as seguintes propriedades são válidas:

- valores literais podem ser escritos entre aspas simples ou duplas (‘abc’ e “abc”)
- o operador + concatena strings (“a” + “a” = “aa”)
- o operador *, quando aplicado a um int n e a uma string s, dá origem à string composta por s concatenada n vezes (“a” * 3 = “aaa”)
- o operador * não pode ser aplicado a dois operandos de tipo string (“a” * “a” é um erro semântico)
- O comprimento de uma string pode ser encontrado pela função **len**. Exemplo: len (“abc”) resulta em 3.
- strings são indexadas a partir do zero e para tal usa-se o operador [].

- a operação de slice (fatiar) permite a obtenção de substrings.

A Tabela 1.7 exibe exemplos de indexação e fatiamento de strings.

Tabela 1.7

Código	Resultado
exemplo = “abcd”	
exemplo[0]	“a”
exemplo[2]	“c”
exemplo[4]	erro (índice fora dos limites)
exemplo[0:2]	“ab”
exemplo[1:2]	b
exemplo[2:2]	“”
exemplo[0:]	“abcd”
exemplo[0:4]	“abcd”
exemplo[0:len(exemplo)]	“abcd”
exemplo[:3]	“abc”
exemplo[-4]	“a”
exemplo[-4:]	“abcd”
exemplo[2:-1]	“c”
exemplo[2:-3]	“”

1.6 Em Python, a **entrada e saída** de dados podem ser realizadas com as funções input e print, respectivamente. A função input devolve str. As funções **int** e **float** permitem a conversão. A Listagem 1.4 mostra exemplos de uso.

Professor Rodrigo Bossini
https://sites.google.com/site/professorrodrigobossini/

Listagem 1.4

```
print (“a”)
print (“a”, “b”)
print (“a”, 1, 2, “n”)
a = input (“Digite uma string”)
inteiro = int (input (“Digite um inteiro”))
real = float (input (“Digite um real”))
```

Exercícios resolvidos pelo professor na aula

1. Escreva um programa que exibe a mensagem “Hello, Python 3”.

2. Escreva um programa que exibe a soma de 3 valores inteiros digitados pelo usuário.

3. Escreva um programa que resolva o mesmo problema do exercício 2 em uma linha só.

4. Escreva um programa que mostra ao usuário seu novo salário, dado que este sofreu um aumento de 32%.

5. Escreva um programa que obtém os coeficientes a, b e c de uma equação do segundo grau e exibe as raizes, a raiz ou uma mensagem que informa que não há raiz.

6. Escreva um programa que obtém o ano de nascimento de uma pessoa e mostra

6.1 Sua idade (aproximada) atual
6.2 Quantos dias (aproximadamente) essa pessoa viveu
6.3 Quantos anos ela terá no ano de 2052
6.4 Quantos anos ela terá em um ano arbitrário que informar

7. Pedro comprou um saco de ração com peso em quilos. Ele possui dois gatos, para os quais fornece a quantidade de ração em gramas. A quantidade diária de ração fornecida para cada gato é sempre a mesma. Faça um programa que receba o peso do saco de ração e a quantidade de ração fornecida para cada gato, calcule e mostre quanto restará de ração no saco após cinco dias.

Bibliografia

[1] PEP 373 -- Python 2.7 Release Schedule. 2018. Disponível em < https://legacy.python.org/dev/peps/pep-0373/>. Acesso em janeiro de 2019.

[2] GUTTAG, J. V. **Introduction to Computation and Programming Using Python**. 1st. ed. MIT Press, 2013.