

# PYTHON PARA JORNALISTAS DE DADOS

## Variáveis, tipos de dados e operações aritméticas

Rodolfo Viana  
MBA em Jornalismo de Dados  
17 de julho de 2021



# Variáveis | Guardando valores na memória


Anteriormente, imprimimos textos com a função `print()`.



```
print("Estamos na segunda aula!")
```

Mas e se a gente quiser guardar a informação para não ter de repetir sempre que quiser usá-la? Nesses casos usamos **variável**.

**Variável** é um nome que se refere a um valor e fica temporariamente salvo na memória do computador. É muito útil, pois sempre que chamarmos a variável, o valor será evocado.



```
# Este bloco traz o mesmo output...  
frase = "Estamos na segunda aula!"  
print(frase)  
  
# ...que esta linha  
print("Estamos na segunda aula!")
```

A atribuição é simples: `variavel = valor`.

## Variáveis | Exemplo

Vamos pegar o primeiro parágrafo de "Alice no País das Maravilhas":

```
alice = 'Alice estava começando a ficar muito  
cansada de estar sentada ao lado de sua irmã e não  
ter nada para fazer: uma vez ou duas ela dava uma  
olhadinha no livro que a irmã lia, mas não havia  
figuras ou diálogos nele e "para que serve um  
livro", pensou Alice, "sem figuras nem diálogos?"'
```

Agora, todas as vezes que eu quiser imprimir o parágrafo, não preciso digitá-lo: basta chamar a variável `alice`:

```
print(alice)
```

# Variáveis | Observações

Há alguns alertas sobre o uso de variáveis:

a variável se perde quando encerrado o script

a variável pode ter o valor sobrescrito

a variável pode ser feita de letras, *underscore* (`_`) ou, no meio ou no fim, números

a variável não pode conter acentos ou pontos

a variável diferencia maiúsculas e minúsculas

a variável não pode ser uma palavra-chave de Python: `break`, `class`, `continue`, `finally`, `global`, `True`, `False` etc. [[ver mais aqui](#)].

# Variáveis | Exemplos

```
# a variável pode ter o valor sobrescrito
```

```
nome = 'Ana'
```

```
print(nome) # OUTPUT: Ana
```

```
nome = 'João'
```

```
print(nome) # OUTPUT: João
```

```
# a variável pode ser feita de letras, underscore (_) ou, no meio ou no fim, números
```

```
idade_1 = 40 # correto
```

```
1_idade = 40 # errado
```

```
# a variável não pode conter acentos ou pontos
```

```
municipio = 'São Paulo' # correto
```

```
município = 'São Paulo' # errado
```

```
# a variável diferencia maiúsculas e minúsculas
```

```
nome = 'Pedro'
```

```
Nome = 'Maria'
```

```
print(Nome) # OUTPUT: Maria
```

```
# a variável não pode ser uma palavra-chave
```

```
class1 = 12.7 # correto
```

```
class = 12.7 # errado
```

# Tipos de dados | Números, letras e quase todo o universo

Reparem que, nos exemplos anteriores, foram usadas variáveis com valores com e sem aspas, com e sem pontos...

Isso se explica da seguinte forma: os valores são de **tipos** diferentes.

Python tem muitos **tipos**, cada um com suas características. ➡

```
# NUMÉRICOS (não usam aspas)

# Integer (int): números inteiros
exemplo_1 = 7
exemplo_2 = 63421

# Float (float): números decimais, com ponto flutuante; notação científica
exemplo_3 = 12.15
exemplo_4 = 0.0008
exemplo_5 = 1e-30 # 30 zeros antes de 1

# Boolean (bool): comporta dois valores: True (verdadeiro, 1) e False (falso, 0)
exemplo_6 = True
exemplo_7 = False

# NÃO-NUMÉRICO (usam aspas duplas ou simples)

# String (str): letra, texto, sequência de caracteres alfanuméricos
exemplo_8 = 'a'
exemplo_9 = "São Paulo"
exemplo_10 = "1a2b3c"
exemplo_11 = '63421'
```

# Tipos de dados | Vamos brincar com tipos

Vamos colocar a mão na massa:

1. Abra o VS Code
2. Digite algumas linhas

Nesta atividade vamos usar as funções:

- `print()`: exibe na tela os objetos [[doc](#)]
- `type()`: retorna o tipo do objeto [[doc](#)]
- `str()`: converte o objeto para *string* [[doc](#)]
- `int()`: converte o objeto para *integer* [[doc](#)]
- `float()`: converte o objeto para *float* [[doc](#)]

```
temperatura = 27.6
print(temperatura)
print(type(temperatura))
```

```
salario = 1100
print(salario)
print(type(salario))
```

```
salario = '1100'
print(salario)
print(type(salario))
```

```
restaurante = "McDonald's"
print(restaurante)
print(type(restaurante))
```

# Tipos de dados | Vamos brincar com tipos

Agora vamos mudar os tipos!

```
temperatura = 27.6  
print(temperatura)  
print(type(temperatura))
```

```
temperatura_txt = str(temperatura)  
print(temperatura_txt)  
print(type(temperatura_txt))
```

```
temperatura_int = int(temperatura)  
print(temperatura_int)  
print(type(temperatura_int))
```

```
salario = 1100  
print(salario)  
print(type(salario))
```

```
salario_float = float(salario)  
print(salario_float)  
print(type(salario_float))
```

```
salario_txt = str(salario)  
print(salario_txt)  
print(type(salario_txt))
```

Experimente: o que acontece se eu converter `str` para `int` ou `float`?



# Tipos de dados | Cuidado com strings!

`str` requer aspas, que podem ser simples ou duplas.

Mas é preciso uniformidade: se usamos aspas simples no começo, devemos usar aspas simples no fim; o mesmo vale para aspas duplas. Caso contrário, encontraremos erro.

```
nome = 'Rodolfo Viana'
```

```
nome = "Rodolfo Viana"
```

Além disso, quando a `str` contém aspas em si, é necessário:

1. diferenciar as aspas, ou
2. usar *escape* (\) antes das aspas literais, que diz ao sistema para interpretar as aspas literalmente

```
frase = 'Ela disse "bom dia"'
```

```
frase = "Ela disse \"bom dia\""
```

# Operações aritméticas | Primeiros cálculos

Agora que sabemos o que são variáveis e conhecemos os tipos básicos, podemos juntar as duas coisas para resolver **operações matemáticas**.

Como descobrir o valor do salário mínimo por dia ou calcular a variação do salário mínimo entre os anos anterior e atual.

```
sal_minimo = 1100
dias_uteis = 22
print(sal_minimo / dias_uteis)
```

```
# OUTPUT
# 50.0
```

```
sal_2021 = 1100
sal_2020 = 1045
diferenca = sal_2021 - sal_2020
variacao = diferenca / sal_2020
print(variacao)
```

```
# OUTPUT
# 0.05263157894736842
```

# Operações aritméticas | Operadores

Em Python, para fazer cálculos matemáticos, os operadores são:

- adição: `+`
- subtração: `-`
- multiplicação: `*`
- divisão: `/`
- exponenciação: `**`
- parte inteira (descarta decimais): `//`
- módulo (o resto de uma divisão): `%`

A ordem de execução de operações segue a ordem matemática:

1. exponenciação
2. multiplicação e divisão
3. soma e subtração

# Operações aritméticas | Exercício 1

Sem rodar código algum e considerando as variáveis `quatro = 7` e `zero = 2`, diga: qual o output de cada linha?

```
1 print(quatro + zero)
2 print(quatro / zero)
3 print(quatro // zero)
4 print(quatro % zero)
5 print(quatro * zero - quatro / zero)
6 print(quatro * (zero - quatro) / zero)
```

## Operações aritméticas | Exercício 2

Segundo o G1 em 9 de julho [[fonte](#)], até aquela data 82.908.617 pessoas haviam tomado a primeira dose da vacina contra a covid-19. Especificamente naquele dia, 994.468 pessoas tomaram a primeira dose.

Arredondando, o Brasil tem 212 milhões de habitantes, dos quais cerca de 21% tem menos de 18 anos -- ou seja, não são elegíveis para a vacinação.

1. Quantos brasileiros são elegíveis para a vacinação?
2. Se o ritmo de vacinação da primeira dose se mantiver como no dia 9 de julho, em quantos dias (partindo do dia posterior, dia 10) toda a população elegível terá recebido a primeira dose?

## Entenda mais | Material complementar

### Para assistir

- *Nomenclatura das variáveis em Python*, em eXcript [[link](#)]
- *Conhecendo tipos de dados*, em Programação Dinâmica [[link](#)]
- *Tipos de dados*, em Procópio na Rede [[link](#)]
- *Operadores e expressões aritméticas*, em Bóson Treinamentos [[link](#)]

### Para ler

- *Python variables*, em Real Python [inglês] [[link](#)]
- *Operadores*, em Algoritmos em Python [[link](#)]
- *Python data types*, em Real Python [inglês] [[link](#)]

## Tarefa | Atividade para casa

No Canvas há alguns arquivos chamados notebook. Os arquivos são todos iguais, contêm as mesmas informações; a diferença é que estão em formatos diferentes: pdf para ler em qualquer lugar, html para ler no browser e ipynb para ler no Jupyter Notebook.

Ao fim dos arquivos estão as instruções para o exercício.

Vocês precisam compreender o pedido de análise e responder as perguntas por meio de cálculos em Python.

Atividade	Entrega	Nota
análise	27/07	0,2
Enviar arquivo py para eu@rodolfoviana.com.br		