



escola
britânica de
artes criativas
& tecnologia

Python para análise de dados



VARIÁVEIS E TIPOS DE DADOS



GUIA DA AULA 4



Conheça as strings



Acompanhe aqui
os temas que
serão tratados
na videoaula

● **Motivação**

● **Definição**

● **Operações**

● **Métodos**

● **Conversão**

● **Revisitando a
motivação**



1. Motivação

A empresa que você trabalha adquiriu uma startup de logística. Você precisa identificar todos endereços que são comum a ambas. Na sua empresa, você armazena a latitude e longitude dos endereços em duas variáveis `lat` e `lon`, já a *startup* adquirida em uma única variável `latlon`.

```

In [ ]:
# sua empresa
lat = '-22.005320'
lon = '-47.891040'

# startup adquirida
latlon = '-22.005320;-47.891040'
  
```

Como podemos normalizar a forma com que as latitudes e longitudes são armazenadas para que possam ser comparadas?



2. Definição

Armazenam **textos**:

c , EBAC , Andre Perez, 20 anos (texto)

São do tipo `str`

In []:

```
nome_aula = 'Aula 04, Módulo 01, Strings'

print(nome_aula)
print(type(nome_aula))
```

In []:

```
string_vazia = ""

print(string_vazia)
print(type(string_vazia))
```



3. Operações

As operações de variáveis do tipo *string* são:

+ (concatenação);

Exemplo: Nome completo

In []:

```

nome = 'Andre Marcos'
sobrenome = 'Perez'

apresentacao = 'Olá, meu nome é ' + nome + ' ' + sobrenome + ' .'
print(apresentacao)

```



3. Operações

Uma outra forma de concatenar *strings* é utilizar operações de formatação:

```

In [ ]:
nome = 'Andre Marcos'
sobrenome = 'Perez'

apresentacao = f'Olá, meu nome é {nome} {sobrenome}.'
print(apresentacao)
  
```

Outra operação muito utilizada é a de fatiamento (*slicing*):

Exemplo: Informações de e-mail.

```

In [ ]:
email = 'andre.perez@gmail.com'
  
```



3. Operações

Fatiamento fixo:

a	n	d	r	e	.	a		p	e	r	e	z	@	g	m	o	m			
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

In []:

```
print('0: ' + email[0])
print('11: ' + email[11])
```

In []:

```
print('-1: ' + email[-1])
print('-2: ' + email[-2])
```



3. Operações

Fatiamento por intervalo:

a	n	d				r	e	.				p	e		r	e		z		@		g		m
a	i					l		.				c			o			m						
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20				

```
In [ ]: email_usuario = email[0:11]
        print(email_usuario)
```

```
In [ ]: email_provedor = email[12:21]
        print(email_provedor)
```



4. Métodos

São métodos nativos do Python que nos ajudam a trabalhar no dia a dia com *strings*.

```
In [ ]: endereco = 'Avenida Paulista, 1811, São Paulo, São Paulo, Brasil.'
```

```
In [ ]: # maiusculo: string.upper()
        print(endereco.upper())
```

```
In [ ]: # posicao: string.find(substring)
        posicao = endereco.find('Brasil')
        print(posicao)
```

```
In [ ]: # substituição: string.replace(antigo, novo)
        print(endereco.replace('Avenida', 'Av'))
```



5. Conversão

Podemos converter *strings* em tipos numéricos e vice-versa.

```

In [ ]:
idade = 19
print(type(idade))

idade = str(idade)
print(type(idade))
  
```

```

In [ ]:
faturamento = 'R$ 35 mi'
print(faturamento)
print(type(idade))

faturamento = int(faturamento[3:5])
print(faturamento)
print(type(faturamento))
  
```



6. Revisitando a motivação

Encontrando a posição do caracter ; de divisão das *strings* de latitude e longitude da variável da *startup*:

```
In [ ]: posicao_char_divisao = latlon.find(';')
        print(posicao_char_divisao)
```

Extraindo a latitude:

```
In [ ]: lat_startup = latlon[0:posicao_char_divisao]
        print(lat_startup)
```

Extraindo a longitude:

```
In [ ]: lon_startup = latlon[posicao_char_divisao+1:len(latlon)]
        print(lon_startup)
```

