4. *Strings*

4.1. 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`.

```
# 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 possam ser comparadas?

4.2. Definição

Armazenam **textos**:

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

São do tipo `str`:

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

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

```
Aula 04, Módulo 01, Strings
<class 'str'>
```

```
string_vazia = ""

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

```
<class 'str'>
```

4.3. Operações

As operações de variáveis do tipo *string* são:

- `+` (concatenação).

Exemplo: Nome completo.

```
nome = 'Andre Marcos'
sobrenome = 'Perez'

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

```
Olá, meu nome é Andre Marcos Perez.
```

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

```
nome = 'Andre Marcos'
sobrenome = 'Perez'

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

```
Olá, meu nome é Andre Marcos Perez.
```

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

Exemplo: Informações de email.

```
email = 'andre.perez@gmail.com'
```

Fatiamento fixo:

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

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

```
0: a
11: @
```

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

```
-1: m
-2: o
```

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

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

```
andre.perez
```

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

```
gmail.com
```

4.4. Métodos

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

```
endereco = 'Avenida Paulista, 1811, São Paulo, São Paulo, Brasil.'

# maiusculo: string.upper()
print(endereco.upper())
```

```
AVENIDA PAULISTA, 1811, SÃO PAULO, SÃO PAULO, BRASIL.
```

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

```
46
```

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

```
Av Paulista, 1811, São Paulo, São Paulo, Brasil.
```