Strings

Strings ou str(s) é um tipo de dados diferenciado, apesar de parecerem simples, a implementação do tipo str é bastante complexo, porém ao mesmo tempo permite que diversas operações possam ser feitas com esse tipo de dados.

Nesta aula, vamos ver apenas algumas das operações que podem ser feitas utilizando strings, vamos lá.

A primeira operação que pode ser efetuada com strings é a concatenação, que consiste em "somar" duas strings diferentes.

```
nome = "Gabriel"
sobrenome = "Azevedo"
print(nome + sobrenome)
# Gabriel Azevedo
```

A concatenação pode ser feita também com strings e outros tipos também como int e float, mas para isso precisamos utilizar um método de formatação, que convenientemente é chamado de format().

```
txt = "Pi é {}"
pi = 3.14
print(txt.format(pi))
```

O método format substitui instancias de "{}" por argumentos da função format, podemos utilizar mais de uma variável neste método, conforme o exemplo abaixo.

```
txt = "Pi é {} e x é {}"
pi = 3.14
x = 4
print(txt.format(pi, x))
```

É possível ainda substituir valores em uma posição arbitrária, para isso, basta informar a posição do valor na ordem dos argumentos.

```
txt = "Pi é {1} e x é {0}"
pi = 3.14
x = 4
print(txt.format(x, pi))
```

Como você deve ter visto, os valores na lista de argumentos da função format (e de qualquer outra função em python) sempre iniciam em zero, no exemplo acima, {0} corresponde a x e {1} corresponde a pi.

As strings em python são sempre consideradas como uma sequencia de caracteres individuais, sendo que cada caractere ocupa uma posição única na lista, outro ponto importante é que não existem espaços vazios nesta lista, e até espaços são considerados elementos desta lista.

 Mais à frente veremos o conceito de Listas em python, que vai ajudar bastante a trabalhar com strings de maneira ainda mais dinâmica.

Pode-se acessar um elemento dessa lista utilizando a sintaxe identificador[índice], onde:

- Identificador é o nome da variável que possui o valor em tipo string;
- índice é a posição do caractere que queremos acessar na lista, iniciando em zero (0);

Vamos ver alguns exemplos, considere o texto abaixo:

```
txt = "Olá!"
```

Para acessar a letra "O", podemos utilizar:

```
txt[0] # "O"
```

Ainda é possível saber o tamanho de uma string utilizando o método len():

```
len(txt) # 4
```