**Uso de operadores aritméticos com strings**

# Dois operadores aritméticos possuem um comportamento especial em strings:

# Operador de soma (+): concatena (une) 2 strings.

palavra1 = "Let's"

palavra2 = "Code"

palavra3 = palavra1 + palavra2

print(palavra3)

# Operador de multiplicação (\*): copia uma string 'n' vezes:

palavra = 'uma'

trespalavras = 3\*palavra

print(trespalavras)

**Formatação de strings**

Uma última função interessante de strings é o .format() Para entender como ela funciona, podemos pensar em um contrato. É normal que ele venha com campos em branco para serem preenchidos posteriormente. Ex:

Eu, \_\_\_\_\_\_\_\_\_\_\_\_\_\_**, portador do RG \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ e nascido ao dia //**, autorizo o uso de minha imagem.

Ao inserirmos no contrato nossas informações pessoais, ele se torna completo.

O .format() serve para "preencher" os "campos em branco" de uma string.

Os campos em branco serão representados por pares de chave: {}

prod = 'chocolate'

preco = 3.14

print('O produto {} custa {}.'.format(prod, preco))

# Na linha acima, prod substituirá o primeiro {}, e preco o segundo {}.

# Saída: O produto chocolate custa 3.14.

# É possível colocar algumas opções especiais de formatação. Por exemplo:

stringoriginal = 'O litro da gasolina custa {:.2f}'

'''

: indica que passaremos opções

.2 indica apenas 2 casas após o ponto decimal

f indica que é um float

'''

preco = 3.14159265

stringcompleta = stringoriginal.format(preco)

print(stringcompleta)

# Saída: O litro da gasolina custa 3.14

# O preço sai com apenas 2 casas após a vírgula!

# Pode-se chamar as variávies em diferentes ordens:

print('{2}, {1} and {0}'.format('a', 'b', 'c'))

# Saída: c, b and a

print('{0}{1}{0}'.format('abra', 'cad'))

# Saída: abracadabra

# Podemos especificar um número de dígitos obrigatório por campo.

# Vejamos o exemplo:

dia = 1

mes = 8

ano = 2019

data1 = '{}/{}/{}'.format(dia, mes, ano)

print(data1)

# Saída: 1/8/2019

# O dia e o mês ocupam apenas 1 espaço

data2 = '{:2d}/{:2d}/{:4d}'.format(dia, mes, ano)

# A opção 'd' significa que o parâmetro é um número inteiro.

print(data2)

# Saída: 1/ 8/2019

# Agora, dia e mês ocupam obrigatoriamente 2 espaços: 1/ 8/2019

# Podemos forçar que os espaços em branco sejam preenchidos com o número 0:

data3 = '{:02d}/{:02d}/{:04d}'.format(dia, mes, ano)

print(data3)

# Saída: 01/08/2019

# Agora sim a data está no formato usual, dd/mm/aaaa: 01/08/2019

# Um modo mais simples de utilizar o format

nome = "Pietro"

profissao = "professor"

escola = "Let's Code"

print(f"{nome} é {profissao} da {escola}.")

# Saída: Pietro é professor da Let's Code.