Lógica de Programação Orientada a Objetos

Bem-vindos!

Objetivos

- Strings
- Arquivos

LET'S CODE

STRINGS

- Os seguintes literais de strings são equivalentes:

```
"Hello World!"
'Hello World!'
"""Hello World!"""
'''Hello World!'''
```

- Escolha o tipo de aspas que precisa:

```
"It's a very nice day."
'The sign says "Hello World!".'
```

STRINGS MULTI LINHA

- Aspas triplas habilita o string multi-linha em seu código:

```
"""Shopping List:
Cheese
Apples
Bread"""

'''ABC
DEF
GHI'''
```

SEQUÊNCIAS DE ESCAPE

- \n: nova linha
- \t: tabulação
- \' e \": aspas simples e aspas duplas.
- \\: barra

Exemplo:

```
"I wrote: \"Hello!\".\nHe wrote: \"Goodbye!\"."
```

STRINGS CRUAS (RAW STRINGS)

- Para desabilitar a função de escape, as raw strings podem ser usadas:

```
>>> print('c:\windows\newstuff\todo')  # OOPS!
c:\windows
ewstuff odo
>>> print(r'c:\windows\newstuff\todo') # Better.
c:\windows\newstuff\todo
```

CHECAGEM

- endswith, startswith

```
'hello world'.startswith('he')  # -> True
```

- isalnum, isalpha, isdigit, islower, isupper, isspace

```
'123'.isdigit()  # -> True
'Hello World'.islower()  # -> False
```

LET'S CODE

PESQUISAS

- count

```
'hello world'.count('l')  # -> 3
```

- find

```
'hello world'.find('l')  # -> 2
'hello world'.find('t')  # -> -1
```

MANIPULAÇÕES

Esteja atento, porque str é um tipo imutável. Todos os métodos abaixo retornam uma nova string.

- ✔ lower, upper, title, capitalize, swapcase

```
'hello world'.title()  # -> 'Hello World'
'hello world'.capitalize()  # -> 'Hello world'
```

- ✔ replace

```
'hello world'.replace('world', 'john')  # -> 'hello john'
```

- ✔ strip, rstrip, lstrip - remove espaços e nova linha no fim da string:

```
'     hello!    \n'.strip() # -> 'hello!'
```

+ E *

```
'hello ' + 'world'  # -> 'hello world'

'hello ' * 3  # -> 'hello hello hello '
```

LET'S CODE

FORMATANDO

```
name = 'Tom'

# Estilo de formatação ruim
print('hello ' + name + '!' )

# Estilo de formatação antiga
print('Hello %s!' % name)

# Estilo de formatação nova
print('Hello {}!'.format(name))
```

- Formatação nomeada

```
TMPL = 'You got an error in {file} line {line}'
#.....
# ....

print(TMPL.format(file='a.py', line=5))
```

- Formatação posicionada

```
>>> print('{0} {0}, {1}'.format('repeat me','not me'))
repeat me repeat me not me
```

SLICING

- Similar com parâmetros range
- Índices com pares de colchetes: s[1]
- Índice negativo que retorna o elemento no fim da lista: s[-1]
- Fatias:
    - Sintaxe básica é s[inicio:final]: s[1:4]
    - Fatiando no início da string: s[:5]
    - Fatiando até o final da string: s[3:]
    - Índice negativo também pode ser usado em fatias: s[-3:-1]

```python
s = 'hello world'

# fatiando (incluindo inicio, excluindo o final)
s[1:4]  # -> 'ell'

# a partir do início do índice
s[:4]  # -> 'hell'

# a partir do final do índice
s[3:]  # -> 'lo world'
```

```python
# pode usar índice negativo também
s[2:-2]  # -> 'llo wor'

# saltos
s[::2]  # -> 'hlowrd'
s[1::2]  # -> 'el ol'

# saltos negativos
s[::-1]  # -> 'dlrow olleh'
```

DIVIDINDO, JUNTANDO E LISTANDO

- Dividindo uma string de acordo com o valor:

```
'hello world'.split()  # -> ['hello', 'world']

'hello, and ,welcome'.split(',', maxsplit=1) # -> ['hello', ' and ,welcome']
```

- ['hello', 'world'] é um dado do tipo lista. Nós vamos ver mais sobre isso depois!

- Unindo valores de uma lista usando um separador de string específico

```
' '.join(['hello', 'world'])  # -> 'hello world'

','.join(['first line', 'second line'])  # -> 'first line,second line'
```

ABRINDO ARQUIVOS

- open(name, mode) retorna um objeto do tipo file
- name é o caminho do arquivo a ser aberto
- mode:
  - 'r' (read - leitura): o arquivo é aberto em modo somente leitura
  - 'w' (write - escrita): o arquivo é aberto em modo somente escrita, e é truncado.
  - 'a' (append - adição): como 'e' mas acrescenta no arquivo sem truncar.
  - 'x': como 'w' mas o arquivo não deve existir.
- open(name) padrão para leitura: open(name, 'rt')

FECHANDO

- f.close():
  - Lança um tratamento no arquivo
  - Escreve o conteúdo do objeto file no disco.
- Pode ser feito de forma alternativa usando a declaração with:

```
with open('example.txt') as f:
    print(f.read())
```

LENDO

- f.read() faz a leitura de todo o arquivo (até EOF)
- f.read(index) faz a leitura do arquivo até index

```
# Prints each line of the file.
with open('example.txt') as f:
    for l in f:
        print(l)
```

ESCREVENDO

- f.write(string) escreve string (sem adicionar \n)
- f.writelines(sequence) escreve uma sequência de conteúdo (também sem adicionar \n)

```
fruits = ['Bannana', 'Melon', 'Peach']
with open('example.txt', 'w') as f:
    f.writelines(fruits)
```

Exercícios

- https://www.hackerrank.com/challenges/python-string-split-and-join/problem
- https://www.hackerrank.com/challenges/find-a-string/problem
- https://www.hackerrank.com/challenges/string-validators/problem
- https://www.hackerrank.com/challenges/python-string-formatting/problem
- https://www.hackerrank.com/challenges/capitalize/problem