ARQUIVOS

Comandos de entrada e saída em arquivos

INTRODUÇÃO

- Podem armazenar grande quantidade de informação;
- Dados são persistentes (gravados em disco);
- Arquivos são identificados por um nome;
- O nome de uma arquivo pode conter uma extensão que indica o conteúdo do arquivo, por exemplo:

arquivo.txt	**Arquivo texto**
arquivo.py	Arquivo em Python

TRABALHANDO COM ARQUIVO

- Em Python, devemos **abrir** (*open*) arquivos antes de usá-los e **fechar** (*close*) os arquivos depois de que tivermos terminado de utilizá-los.
- Para abrir um arquivo, o Python possui a função open(). Ela recebe dois parâmetros:
  - o primeiro é o nome do arquivo a ser aberto;
  - o segundo é o modo que queremos trabalhar com esse arquivo - se queremos ler ou escrever.
  - Por exemplo:

```
arquivo = open('listaDeCompras.txt', 'w')

arquivo.close()
```

MODOS DE ABERTURA DE UM ARQUIVO

- ‘r’ → *read*, leitura: usado somente para ler;
- ‘w’→*write*, escrita: usado somente para escrever;
- ‘r+’→ usado para ler e escrever;
- ‘a’→ *append*, acrescentar: usado para acrescentar.

- O modo padrão é o ‘r’ → *read*

ESCRITA EM ARQUIVO

```
arquivo = open('listaDeCompras.txt', 'a')
for i in range(3):
        aux = input("Digite um item: ")
        arquivo.write(aux)
        arquivo.write("\n")
arquivo.close()
```

ESCRITA EM ARQUIVO - INSERINDO QUEBRA DE LINHA

```
arquivo = open('listaDeCompras.txt', 'w')
arquivo.write('refri\n')
arquivo.write('batata\n')
arquivo.write('queijo\n')
arquivo.close()
```

E SE...

- Modo de abertura: escrita
  - E se o arquivo não existir?
    O arquivo é criado.

LEITURA EM ARQUIVO

```
arquivo = open('listaDeCompras.txt', 'r')
for linha in arquivo:
   print(linha)
arquivo.close()
```

E SE...

- Modo de abertura: leitura
  - E se o arquivo não existir?
    - O Python vai lançar o erro FileNotFoundError

INCLUSÃO DE DADOS EM ARQUIVO

```
arquivo = open('listaDeCompras.txt', 'a')
arquivo.write('bacon')
arquivo.close()
```

LENDO LINHA DO ARQUIVO

- Entre cada item há uma linha vazia, visto que a função print() acrescenta, por padrão, um \n
- Para isso, pode-se utilizar a função:

readline()

- Para ler apenas uma linha do arquivo.

```
arquivo = open('listaDeCompras.txt', 'r')
linha = arquivo.readline()
print(linha)
arquivo.close()
```

FUNÇÃO STRIP()

- Para tirar espaços em branco no início e no fim da string, basta utilizar a função strip(), que também remove caracteres especiais, como o \n

```
arquivo = open('listaDeCompras.txt', 'r+')

for linha in arquivo:
        linha = linha.strip()
arquivo.write('\n\nstrip: '+linha+"\n")
arquivo.close()
```