

Arquivos em Python

- Uma das tarefas mais importantes que realizamos no dia-a-dia é a manipulação de dados gravados em **arquivos**;
- Por exemplo, um administrador de sistemas precisa, com frequência, acessar informações de configuração armazenadas em arquivos de texto, e muitas vezes, efetuar alterações nesses arquivos.

Arquivos em Python

- Um **arquivo** é uma área no disco onde gravamos e de onde lemos dados. Um arquivo pode ser de texto simples ou um arquivo binário;
- **Arquivos de texto** (plaintext) são uma sequência estruturada de linhas com sequência de caracteres;
- **Arquivos binários** é qualquer arquivo que não seja arquivo de texto padrão, como pdf, doc, imagens e etc.

Abrindo arquivos em Python

- O primeiro processo natural que temos que realizar para fazer qualquer operação sobre algum arquivo é **abri-lo**;
- Para abrir um arquivo usando Python, usaremos a função **open()**. A função retorna um objeto de arquivo e é mais comumente usado com dois argumentos:

Abrindo arquivos em Python

`open(filename, mode)`



Primeiro argumento(filename)

É simplesmente o nome do arquivo que deseja abrir



Segundo argumento(mode)

É uma string que indica como o arquivo vai ser aberto

Abrindo arquivos em Python

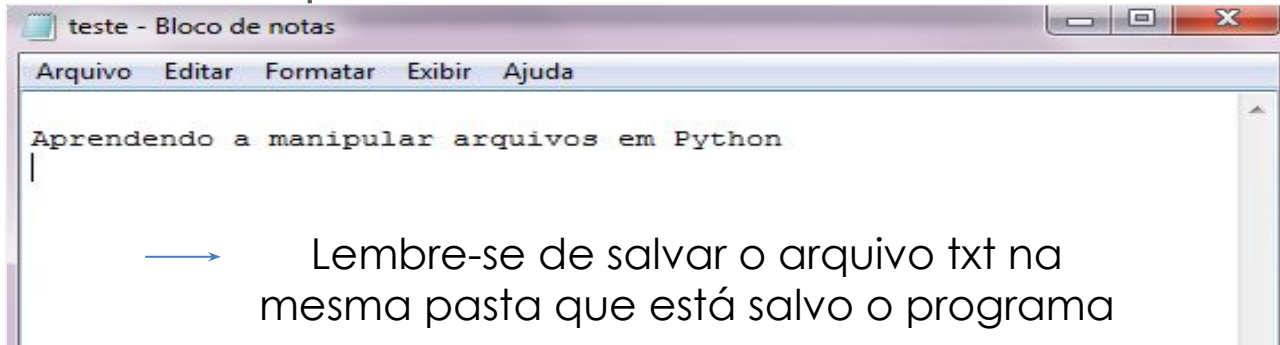
- Exemplos de modos(modes) diferentes para abrir um arquivo são mostrados na tabela a seguir:

Descrição dos modos

r	Abre o arquivo de texto para leitura.
r+	Abre para leitura e escrita.
w	Trunca o arquivo para zero ou cria um arquivo de texto para escrita.
w+	Abre para leitura e escrita. O arquivo é criado se ele não existir, caso contrário será sobrescrito.
a	Abre para escrita. O arquivo é criado caso não exista.
a+	Abre para leitura e escrita. O arquivo é criado se ele não existir.

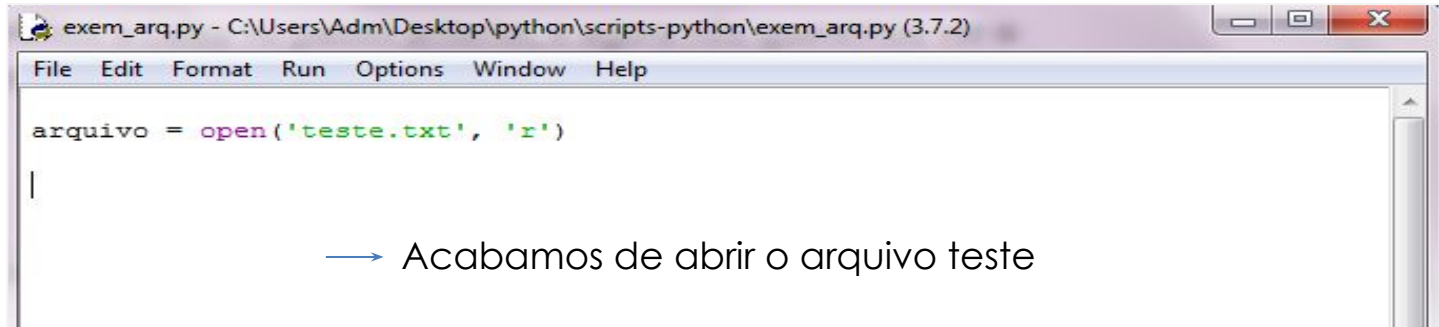
Abrindo arquivos em Python

- Vamos exemplificar:
- Vamos criar um arquivo chamado **teste**



Abrindo arquivos em Python

- Agora abra o arquivo **teste.txt**, isso pode ser feito simplesmente com uma única linha em Python da seguinte maneira:

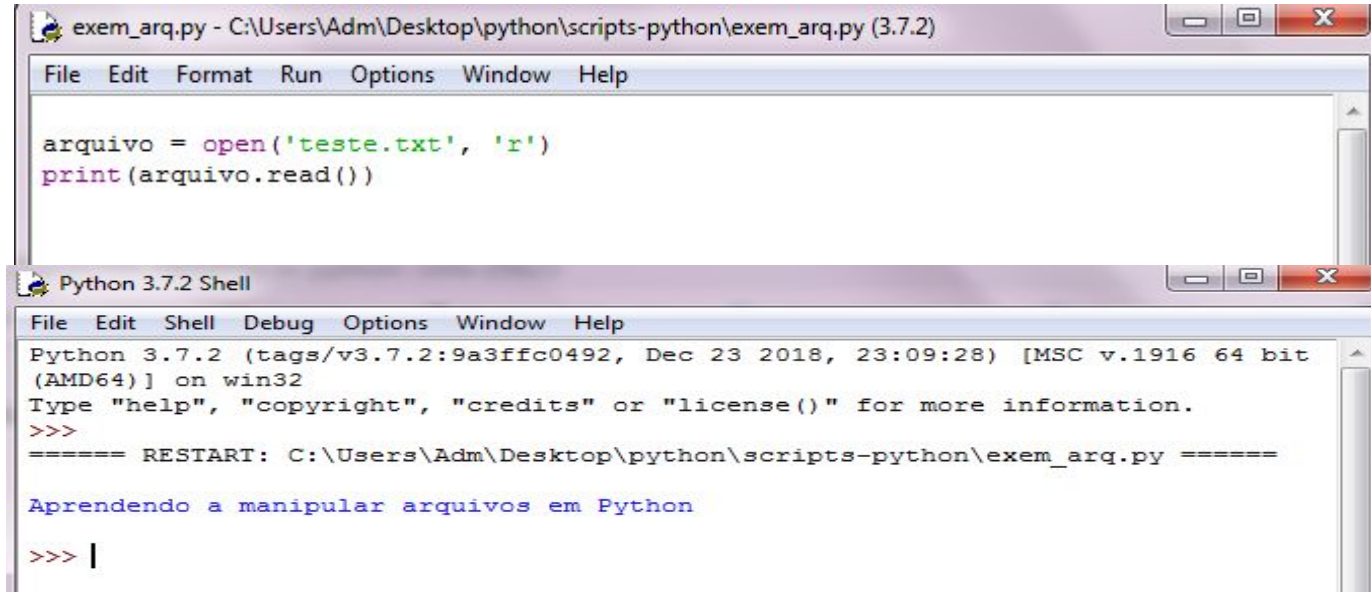


```
exem_arq.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py (3.7.2)
File Edit Format Run Options Window Help
arquivo = open('teste.txt', 'r')
|
→ Acabamos de abrir o arquivo teste
```

Lendo arquivos em Python

- O arquivo é como uma caixa secreta. Nós Abrimos a caixa no passo anterior, e agora nós queremos ver o que há dentro. Ler um arquivo simples em Python pode ser feito usando o método **read()**.
- A função **read()** lê o arquivo todo de uma vez para uma variável. Isso ocorre em casos em que eu quero apenas exibir a variável.

Lendo arquivos em Python



The image shows two overlapping windows from a Python IDE. The top window, titled 'exem_arq.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py (3.7.2)', contains the following Python code:

```
arquivo = open('teste.txt', 'r')
print(arquivo.read())
```

The bottom window, titled 'Python 3.7.2 Shell', shows the execution output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py =====
Aprendendo a manipular arquivos em Python
>>> |
```

Lendo arquivos em Python

Linha por linha

- Além do método `read()`, visto anteriormente, também podemos ler o conteúdo de um arquivo usando os métodos **`readline()`** e **`readlines()`**.

`Readline()`

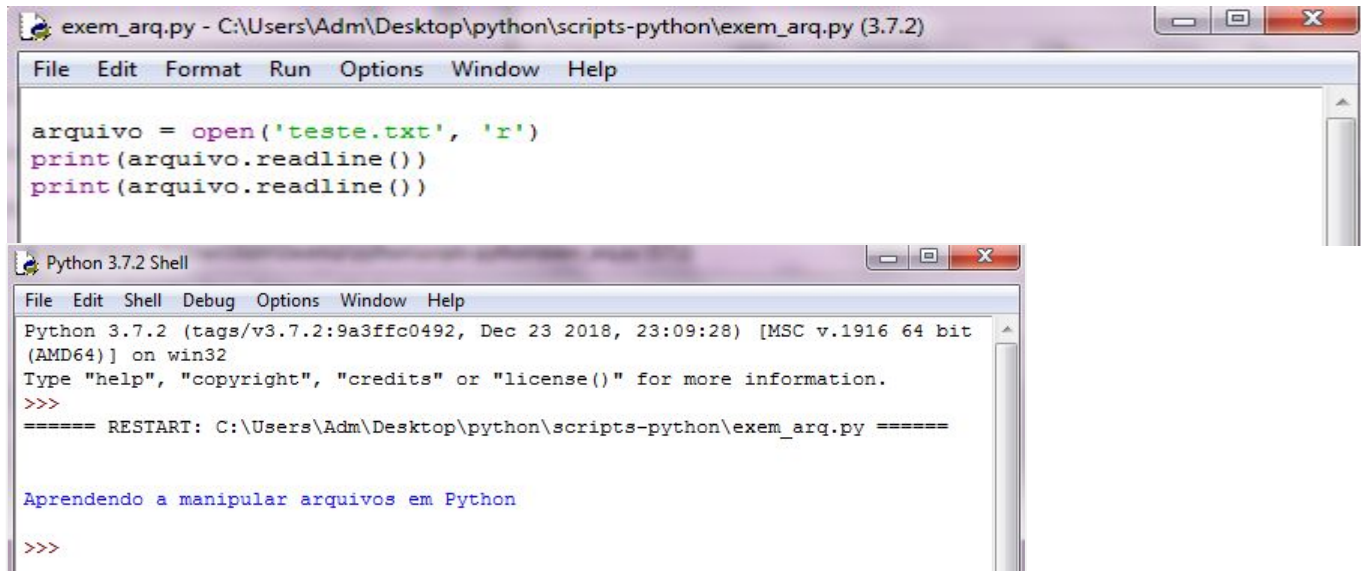
Retorna uma linha do texto a cada chamada, na ordem em que aparecem no arquivo

`Readlines()`

Retorna uma lista de valores de string do arquivo, sendo que cada string corresponde a uma linha do texto.

Lendo arquivos em Python

Linha por linha



The image shows two windows from a Python IDE. The top window, titled 'exem_arq.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py (3.7.2)', contains the following Python code:

```
arquivo = open('teste.txt', 'r')
print(arquivo.readline())
print(arquivo.readline())
```

The bottom window, titled 'Python 3.7.2 Shell', shows the execution output:

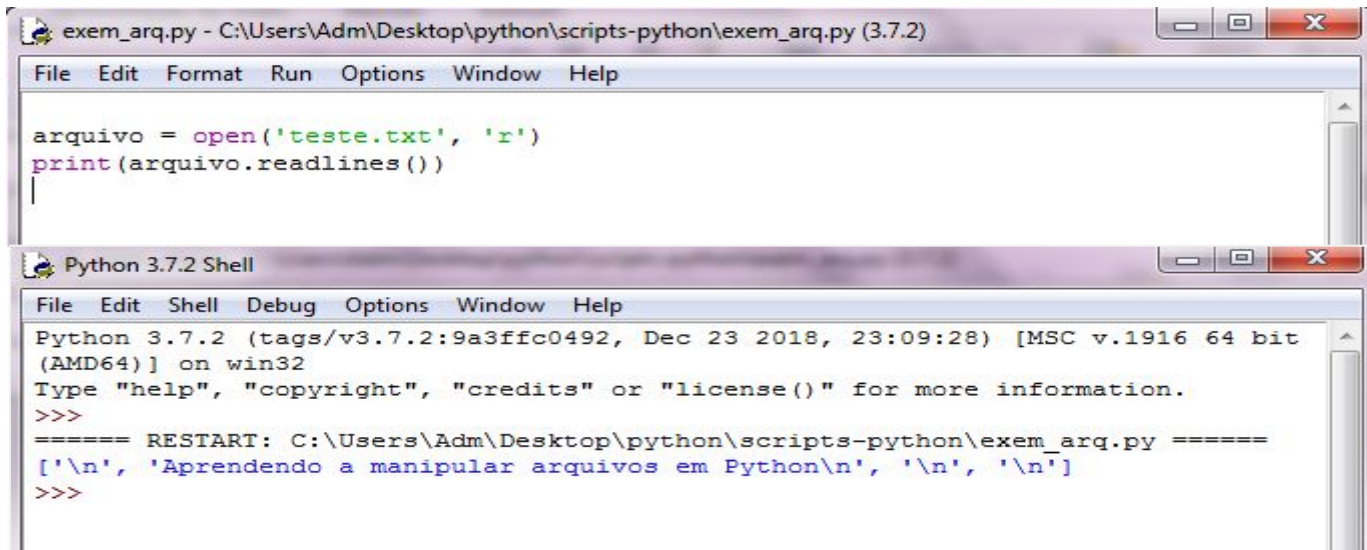
```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py =====

Aprendendo a manipular arquivos em Python

>>>
```

Lendo arquivos em Python

Linha por linha



The image shows two windows from a Python IDE. The top window, titled 'exem_arq.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py (3.7.2)', contains the following Python code:

```
arquivo = open('teste.txt', 'r')
print(arquivo.readlines())
```

The bottom window, titled 'Python 3.7.2 Shell', shows the execution output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Adm\Desktop\python\scripts-python\exem_arq.py =====
['\n', 'Aprendendo a manipular arquivos em Python\n', '\n', '\n']
>>>
```

Escrevendo arquivos em Python

- Talvez você queira escrever outra frase ou parágrafo no arquivo que já lemos. Digamos que queria adicionar a seguinte frase: **Escrevendo em arquivos**. Isso pode ser feito em Python usando o método **write()**;
- A função **write()** utiliza como argumento a **string** que desejamos gravar no nosso arquivo.

Escrevendo arquivos em Python

The image shows a screenshot of a Python IDE window titled "exem_arqui.py - C:/Users/Adm/Desktop/python/scripts-python/exem_arqui.py (3.7.2)". The code in the editor is as follows:

```
arquivo = open('teste.txt', 'w')
arquivo.write('\n Escrevendo em arquivos \n')

arquivo = open('teste.txt', 'r')
print(arquivo.read())
```

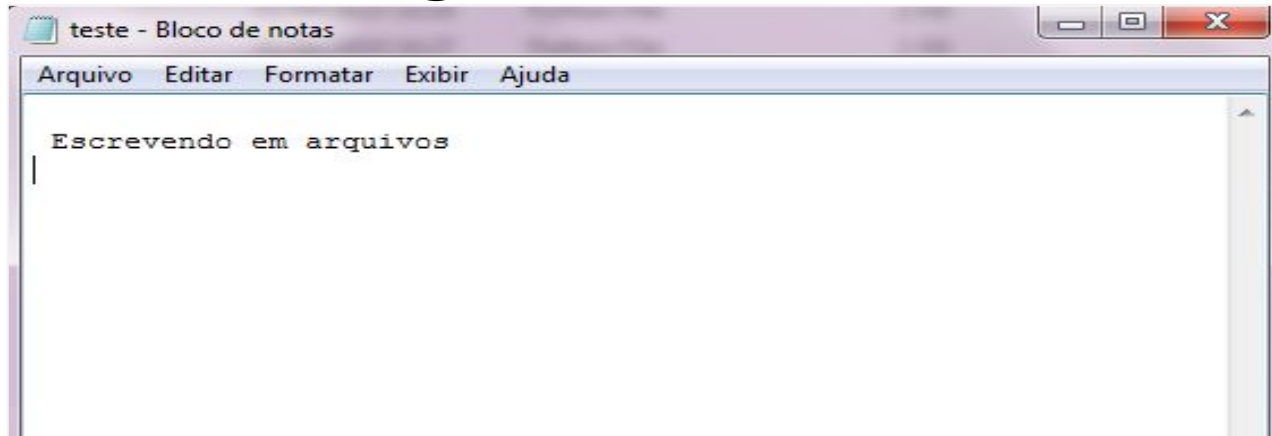
Annotations with arrows point to the file modes:

- An arrow points to the `'w'` mode in the first `open` call, with the text "Modo de gravação w".
- An arrow points to the `'r'` mode in the second `open` call, with the text "Ler o que está gravado dentro do arquivo 'teste'".

Below the code editor is a "Python 3.7.2 Shell" window. It displays the following output:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/Adm/Desktop/python/scripts-python/exem_arqui.py =====
Escrevendo em arquivos
>>> |
```

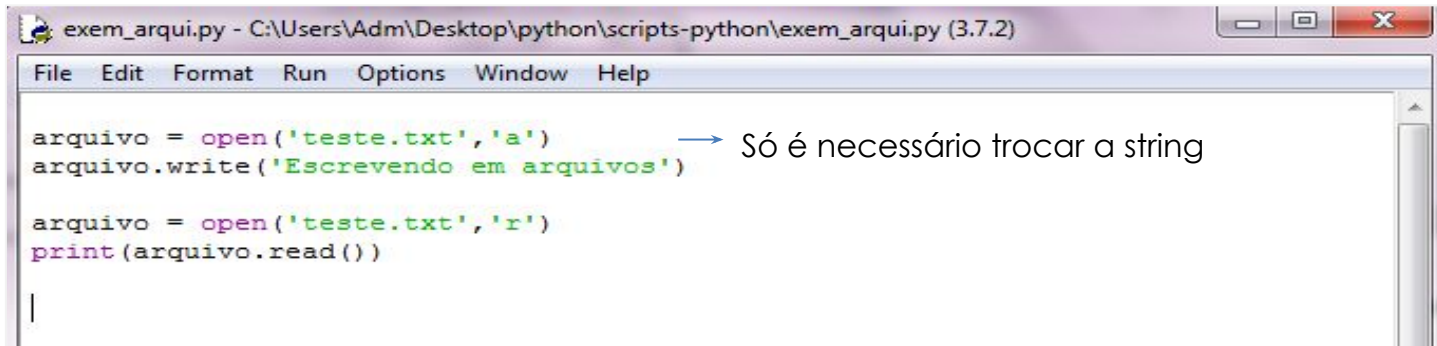
Escrevendo arquivos em Python



Antes o nosso arquivo “**teste**” continha a frase “**Aprendendo a manipular arquivos em Python**” agora o nosso arquivo gravou a nova frase adicionada pelo comando **write**.

Escrevendo arquivos em Python

- Se você está interessado em manter o texto original do documento, você pode usar o modo 'a':



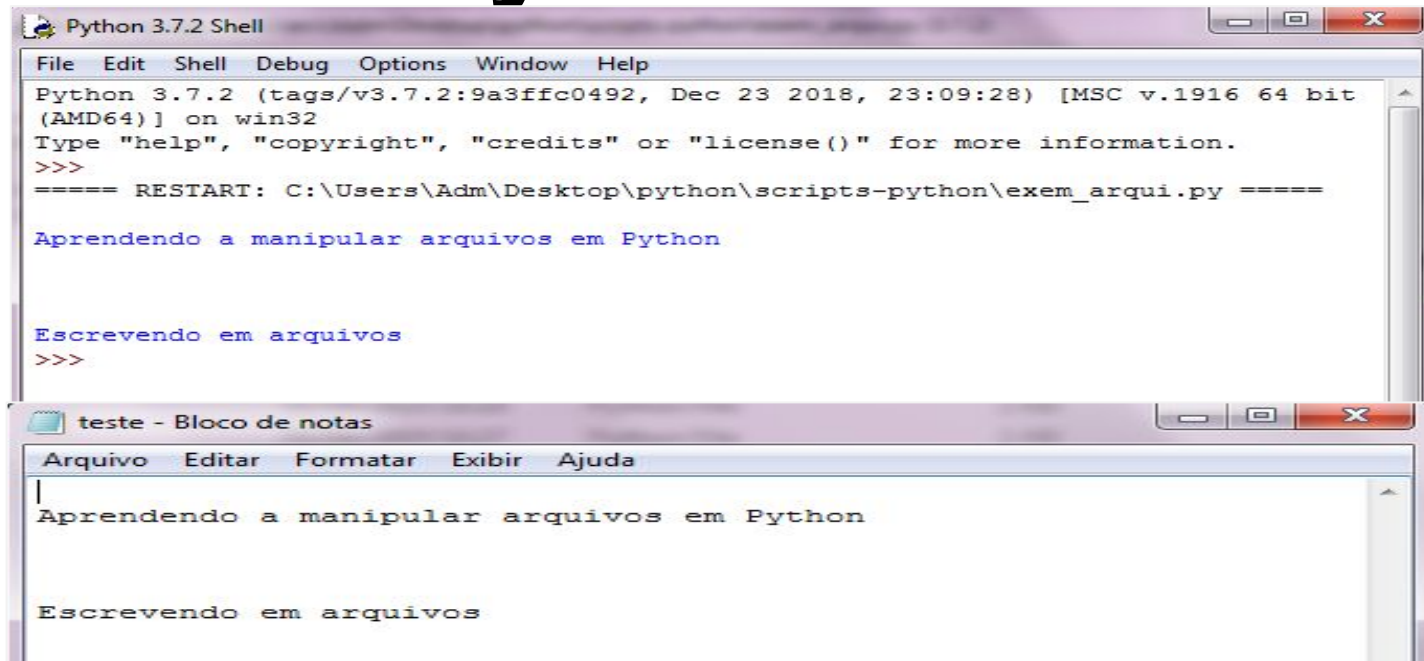
The screenshot shows a Python IDE window titled "exem_arqui.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arqui.py (3.7.2)". The window contains the following Python code:

```
arquivo = open('teste.txt', 'a')
arquivo.write('Escrevendo em arquivos')

arquivo = open('teste.txt', 'r')
print(arquivo.read())
```

An annotation with a blue arrow points to the 'a' mode in the first line, with the text "Só é necessário trocar a string".

Escrevendo arquivos em Python



The image shows two overlapping windows. The top window is titled "Python 3.7.2 Shell" and displays the following text:

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2018, 23:09:28) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\Adm\Desktop\python\scripts-python\exem_arqui.py =====

Aprendendo a manipular arquivos em Python

Escrevendo em arquivos
>>>
```

The bottom window is titled "teste - Bloco de notas" and displays the following text:

```
Arquivo  Editar  Formatar  Exibir  Ajuda

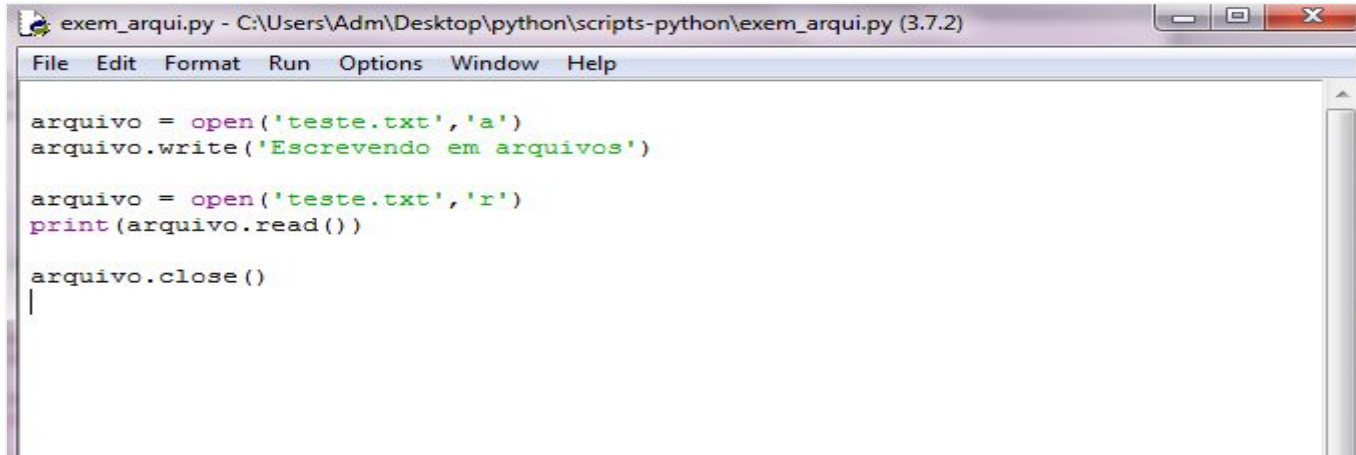
Aprendendo a manipular arquivos em Python

Escrevendo em arquivos
```

Fechando arquivos em Python

- Ter o hábito de fechar um arquivo após a leitura ou escrita permitirá que você libere memória. Sim, o Python fecha automaticamente os arquivos depois que o script termina, mas se você não fizer isso de antemão, todos os arquivos abertos ocuparão espaço que o Python poderia usar.
- Fechar um arquivo pode ser facilmente realizado usando o método **close()**.

Fechando arquivos em Python



```
exem_arqui.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arqui.py (3.7.2)
File Edit Format Run Options Window Help

arquivo = open('teste.txt','a')
arquivo.write('Escrevendo em arquivos')

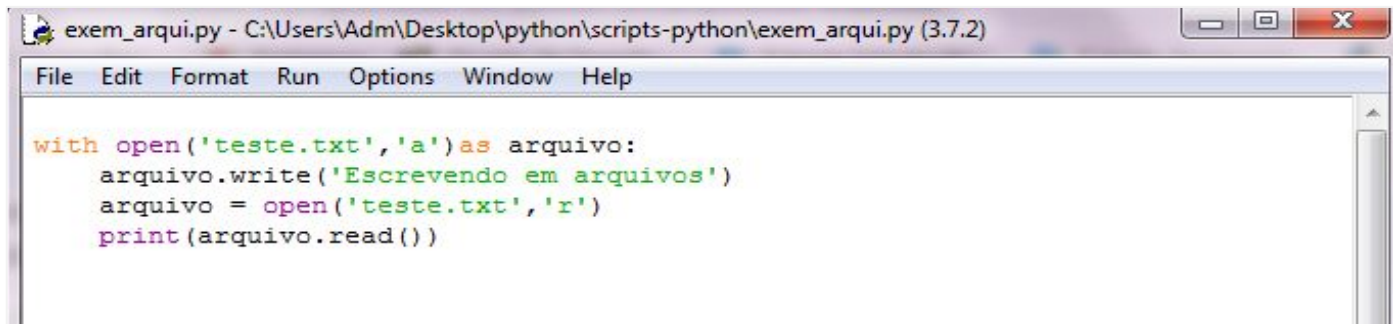
arquivo = open('teste.txt','r')
print(arquivo.read())

arquivo.close()
|
```

Fechando arquivos em Python

- Outra forma de lembrarmos de fechar o arquivo é utilizando o comando **with()**;
- O comando **with()** fecha o arquivo automaticamente assim que saímos de sua suite, ele nos permite abrir o arquivo, processá-lo e certificar-se de que está fechado.

Fechando arquivos em Python



```
exem_arqui.py - C:\Users\Adm\Desktop\python\scripts-python\exem_arqui.py (3.7.2)
File Edit Format Run Options Window Help

with open('teste.txt','a') as arquivo:
    arquivo.write('Escrevendo em arquivos')
arquivo = open('teste.txt','r')
print(arquivo.read())
```

Exercícios Práticos

- Exercício 1:
- ✓ Crie um arquivo chamado “**dados.txt**” e salve seu nome no arquivo e logo depois leia o arquivo.

Exercícios Práticos

- Exercício 2:
- ✓ Com o arquivo criado “**dados.txt**” acrescente a data de nascimento, cidade que nasceu e sua idade e depois leia o arquivo normalmente.

Exercícios Práticos

- Exercício 3:
- ✓ Ainda utilizando o arquivo criado “**dados.txt**” vamos ler o que está dentro do arquivo mas agora utilizando um laço for.

Exercícios Práticos

- Exercício 4:
- ✓ Continuando o exercício anterior, vamos agora contar quantas linhas tem no arquivo **dados.txt**.

Exercícios Práticos

- Exercício 5:
- ✓ Crie um novo arquivo chamado `frases.txt` com as seguintes frases: “**Estudando Python, programando em Python e manipulação de arquivos**” e agora crie um programa que retorne somente as linhas que possuem a palavra **Python** .

Exercícios Práticos

- Exercício 6:
- ✓ Seguindo o exercício anterior, agora vamos criar um programa que retorne somente as linhas que contém uma palavra específica a onde nós iremos dizer qual será a palavra de busca.

Exercícios Práticos

- Exercício 7:

- ✓ Crie um programa que armazene um arquivo chamado **'contatos.txt'** nome e números de pessoas tal como dados a seguir:

Nome	Telefone
Rafael	34567869
Pedro	58762001

Para tanto, crie um laço que leia os nomes e telefones até que seja digitado “sair” quando for requisitado que o nome seja digitado.

Exercícios Práticos

- Exercício 8:
- ✓ Criar um programa que recupera as informações contidas no arquivo '**contatos.txt**' e imprime na tela os nomes e os números de telefone dos contatos.

Exercícios Práticos

- Exercício 9:
- ✓ Crie um programa que armazene em um arquivo chamado **'frota.txt'** o tipo de um navio de guerra, sua posição geográfica representada por duas colunas: latitude(valor entre 0 e 90 graus para o norte ou para o sul) e longitude (valor entre 0 e 180 graus de leste à oeste) e a hora em que foi registrada a informação geográfica tal como dado a seguir:

Exercícios Práticos

- Exercício 9:

Navio	Latitude	Longitude	Hora
Torpedeiro	18 N 20 L		18:02
Submarino	25 N 10 O		18:00
Porta-aviões	20 N 15 L		18:01

- ✓ Para tanto, crie um laço que le o tipo de navio, a latitude, a longitude e a hora até seja digitado “sair” quando for requisitado que o tipo de navio seja digitado.

Exercícios Práticos

- Exercício 10:
- ✓ Criar um programa que recupera as informações contidas no arquivo “**frota.txt**”, e imprime na tela as informações do tipo de navio, latitude, longitude e hora.

Exercícios Práticos

- Exercício 11:
- ✓ Crie um arquivo chamado “**pares.txt**”, com apenas números pares. Logo faça um programa que crie um novo arquivo chamado “**pares_invetidos.txt**”.