

# MANIPULAÇÃO DE ARQUIVOS

Prof. Danilo Ruy Gomes

Última atualização em 18/07/2020

# Introdução

## O que veremos:

- Instruções básicas;
- Escrevendo arquivos;
- Lendo arquivos;
- Alguns exemplos em python

# O que precisaremos

- Python versão 3.7;
- Idle ou Pycharm;
- Bloco de notas

# O que são arquivos

- Basicamente um arquivo é um conjunto de bytes que pode ser escrito ou lido pelo algoritmo;
- Este conjunto de bytes pode ser do tipo binário ou do tipo texto;

# O que são arquivos

## Binários:

- São arquivos gravados/lidos num formato específico e irão armazenar basicamente um conjunto de bytes;
- Caso precisarmos manipular teremos que manipular esse bytes como objetos dentro do arquivo.



# O que são arquivos

## Binários:

- Quando precisamos manipular bytes, normalmente utilizamos bibliotecas específicas para trabalhar esses dados, como é o caso arquivos do Excel, áudio, vídeo, xml, etc.

# O que são arquivos

## Texto:

- São arquivos gravados como uma sequência de caracteres gravados de maneira simples, utilizando a estrutura do padrão ASC do seu S.O.;
- Todos podem ser escritos e lidos por qualquer programa que faça a manipulação de textos.

# Manipulação de arquivos

Método `open()`:

- É o método principal para manipular o arquivo;
- Basicamente ele possui dois parâmetros:
  - O caminho do arquivo, que deve sempre possuir duas barras para separar os diretórios;
  - E a forma como será aberto/escrito, que pode ser leitura, escrita, escrita com leitura, entre outros.



# Datascience - Exemplo

## Exemplo:

O objeto “arquivo” recebe o arquivo que será manipulado

O método “open” possui dois parâmetros



```
exArquivo.py
1 arquivo = open('C:\\Temp\\arq01.txt', 'r')
```

O primeiro parâmetro é o caminho do arquivo

O segundo parâmetro é o tipo de leitura/escrita do arquivo

# Arquivos – comandos básicos

Existem diversos tipos de parâmetro para manipulação de arquivos, eis alguns exemplos:

Método	Descrição
r	Abre o arquivo em modo somente leitura
w	Escrita, sobrescrevendo o conteúdo
a	Escrita, inserindo o novo conteúdo no final do arquivo
b	Modo binário
+	Abre o arquivo para atualização em modo de leitura e escrita
x	Abre o arquivo em modo exclusivo, ou seja, o arquivo não pode ser aberto em dois programas ao mesmo tempo
t	Modo texto. O mais comum.

# Arquivos – comandos básicos

Após carregado, alguns métodos para manipulação do arquivo são:

Método	Descrição
<code>tell()</code>	Posiciona no primeiro byte do arquivo;
<code>read()</code>	Lê o arquivo a partir do byte posicionado
<code>seek(n)</code>	Posiciona o arquivo no byte informado para ser lido a partir deste byte
<code>read(n)</code>	Lê o arquivo somente até o byte informado
<code>readline(n)</code>	Recebe uma linha de texto a cada chamada, posicionando na próxima linha a cada chamada. Caso não for passado parâmetro se iniciará na posição zero

# Arquivos – comandos básicos

Após carregado, alguns métodos para manipulação do arquivo são:

Método	Descrição
<code>readline(n)</code>	Recebe uma linha de texto a cada chamada, posicionando na próxima linha a cada chamada. Caso não for passado parâmetro se iniciará na posição zero
<code>close()</code>	Fecha o arquivo e libera
<code>write</code>	Permite a gravação de uma única informação por vez, por exemplo somente uma variável
<code>writelines()</code>	Permite a gravação de várias informações por vez, por exemplo o conteúdo de um vetor (mais comum)

# Arquivo - Exemplo

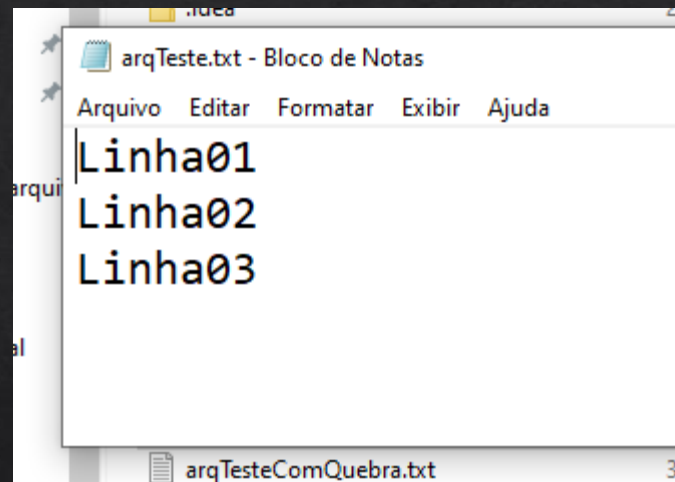
Criando arquivo:

```
def criaArquivo():  
    arquivo = open('C:\\Temp\\arqTeste.txt', 'w')  
    #0 arquivo no objeto "arquivo" será escrito  
    #No caminho c:\\temp com o nome arqTeste.txt  
    #No modo de escrita  
    arquivo.write(str("Linha01\\n"))  
    #Escreva a linha 01  
    arquivo.write(str("Linha02\\n"))  
    # Escreva a linha 02  
    arquivo.write(str("Linha03\\n"))  
    # Escreva a linha 03  
    print("Arquivo criado")  
    arquivo.close()  
    #Fecha o arquivo liberando recursos  
criaArquivo()
```



# Arquivo - Exemplo

O arquivo ficará desta maneira na primeira execução



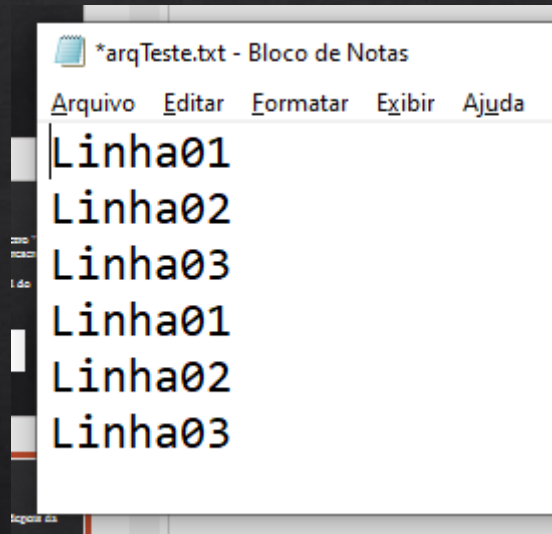
# Arquivo - Exemplo

- Como o arquivo foi aberto com o parâmetro “w”, caso você executar novamente ele irá sobrescrever todo o conteúdo do arquivo.
- Para que ele adicione o conteúdo no final do arquivo altere o parâmetro para “a”

```
def criaArquivo():  
    arquivo = open('C:\\Temp\\arqTeste.txt', 'a')  
    #O arquivo no objeto "arquivo" será escrito  
    #No caminho c:\\temp com o nome arqTeste.txt
```

# Arquivo - Exemplo

- Seu arquivo de texto ficará desta forma, depois da execução



# Arquivo - Exemplo

- Podemos utilizar o método “writelines”, caso quiséssemos passar todo o conteúdo de um vetor, ou outro objeto de uma vez

```
2
3 def criaArquivoComVetor():
4     arquivo = open('C:\\Temp\\arqcomVetor.txt', 'w')
5     arquivo.writelines(("Linha01\n", "Linha02\n", "Linha03\n"))
6     print("Arquivo criado")
7     arquivo.close()
8
```

# Arquivo - Exemplo

Caso fizéssemos o mesmo procedimento com o método “write” teríamos erro na execução, indicando que estes argumentos não são suportados.

```
8
9 def criaArquivo(): #Este método vai dar erro
10     arquivo = open('C:\\Temp\\arqTeste.txt', 'w')
11     arquivo.write(["Linha01\n", "Linha02\n", "Linha03\n"])
12     print("Arquivo criado")
13     arquivo.close()
14
```

exArquivo x

```
File "C:/Users/consultor/PycharmProjects/exArquivos/exArquivo.py", line 11, in criaArquivo
    arquivo.write(["Linha01\n", "Linha02\n", "Linha03\n"])
TypeError: write() argument must be str, not list
```

Process finished with exit code 1



# Arquivo - Exemplo

Podemos criar um arquivo em modo exclusivo, sem permissão de sobreescrita com o parâmetro “x”

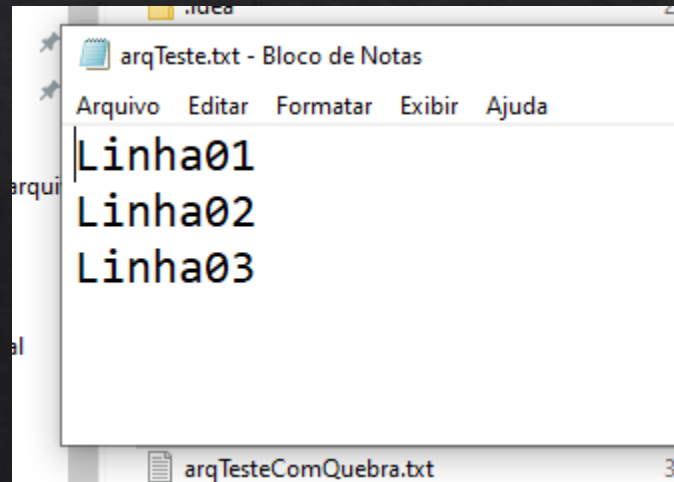
```
def criaArquivoAdicionaEmModoExclusivo():  
    arquivo = open('C:\\Temp\\arqExclusivo.txt', 'x')  
    ##Repare que só mudou o parametro no momento da abertura do arquivo  
    ##Vai dar erro, pois o arquivo já foi criado, assim o parametro "x"  
    ##abre e cria em modo exclusivo, assim ele não sobreescreve o arquivo  
    ##ele só cria o arquivo  
    arquivo.write(str("\nLinha03"))  
    arquivo.close()
```

Tente executar este método novamente que ele não vai deixar criar novamente o arquivo. Pense nos processo do Windows de uso exclusivo.

# Arquivo – Exemplo - Escrita

Alterando linhas a partir de uma determinada posição.

Deixe o arquivo de texto como na imagem:



# Arquivo – Exemplo - Escrita

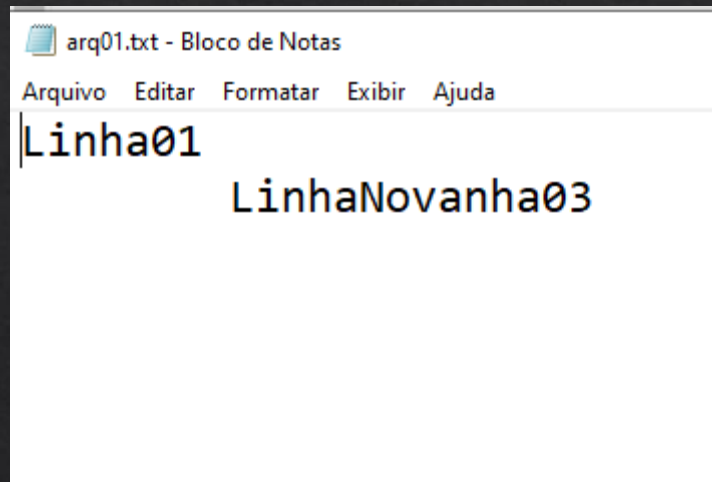
O método abaixo irá sobrescrever o arquivo a partir do “byte 8”, adicionando o caracter asc “tab”.

```
def criaArquivoAlteraLinhas():  
    arquivo = open("C:\\Temp\\arq01.txt", "r+")  
    ##Repare que só mudou o parametro no momento da abertura do arquivo  
    #Abrindo para leitura e escrita  
    arquivo.seek(8)##Posiciona o ponteiro no byte 8 (por isso do r)  
    arquivo.write(str(chr(9)+"LinhaNova"))  
    ##Adicionado o caracter asc tab 9 mais a informacao da linha alterada  
    print("Arquivo alterado a partir do byte 8")  
    arquivo.close()
```

O método “seek” posiciona o cursor na posição 8 e sobrescreve o caractere mais o texto a partir daquela instrução.

# Arquivo – Exemplo - Escrita

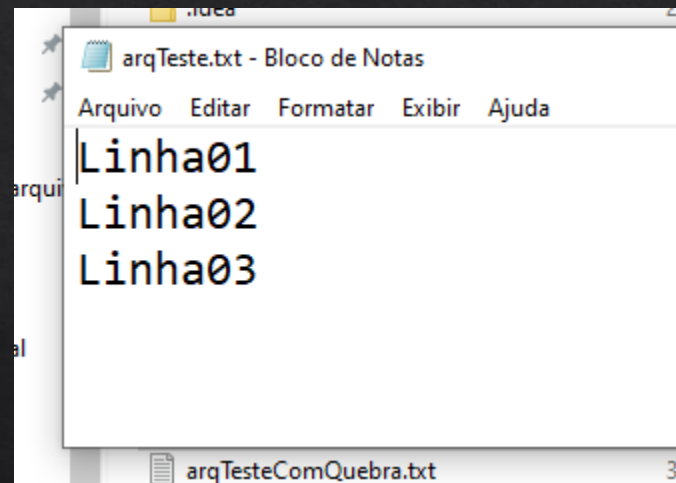
Seu arquivo de texto deverá ficar desta maneira:



Repare que foi sobrescrito a partir do byte 8.

# Arquivo – Exemplo - Leitura

Acerte o arq01.txt para ficar somente com estas linhas:





# Arquivo – Exemplo - Leitura

## Carregando todo arquivo:

O método “tell()”, posiciona a partir do byte inicial e o método read() carrega e apresenta todo o conteúdo do arquivo

```
arquivo = open('C:\\Temp\\arq01.txt', 'r')
```

```
def carregaTodoArquivo():  
    print("Leitura de todo o arquivo posicionado no inicio do arquivo: %s" % arquivo.tell())  
    ##0 tell posiciona no inicio do arquivo por padrao  
    print(arquivo.read()) ##Le os bytes a partir do inicio
```

```
Leitura de todo o arquivo posicionado no inicio do arquivo: 0
```

```
Linha01
```

```
Linha02
```

```
Linha03
```

# Arquivo – Exemplo - Leitura

## Carregando a partir de um determinado byte:

O método “seek”, irá posicionar no byte 10, o “tell()” posiciona a partir do byte inicial 10 e o método read() carrega apresentando todo o conteúdo do arquivo

```
arquivo = open('C:\\Temp\\arq01.txt','r')
```

```
def carregaArquivoAPartirdesteByte():
```

```
    arquivo.seek(10)  ##Faz a leitura a partir deste byte
```

```
    print("Leitura a partir do byte 10: %s" % arquivo.tell())
```

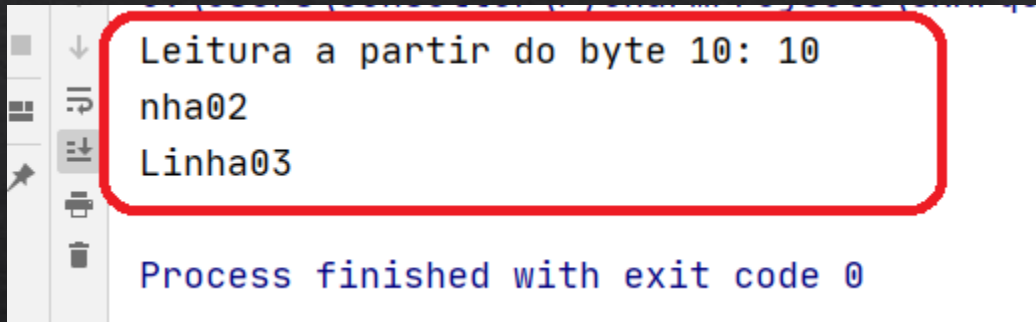
```
    ##Posiciona o cursor a partir deste byte
```

```
    print(arquivo.read())  ##Faz a leitura
```

```
carregaArquivoAPartirdesteByte()
```

# Arquivo – Exemplo - Leitura

O arquivo deverá ser apresentado desta maneira:



A screenshot of a terminal window with a light gray sidebar on the left containing icons for file operations. The main area is white and displays the following text: 'Leitura a partir do byte 10: 10', 'nha02', and 'Linha03'. These three lines are enclosed in a red rounded rectangle. Below this, the text 'Process finished with exit code 0' is displayed in a smaller, blue font.

```
Leitura a partir do byte 10: 10  
nha02  
Linha03  
  
Process finished with exit code 0
```

# Arquivo – Exemplo - Leitura

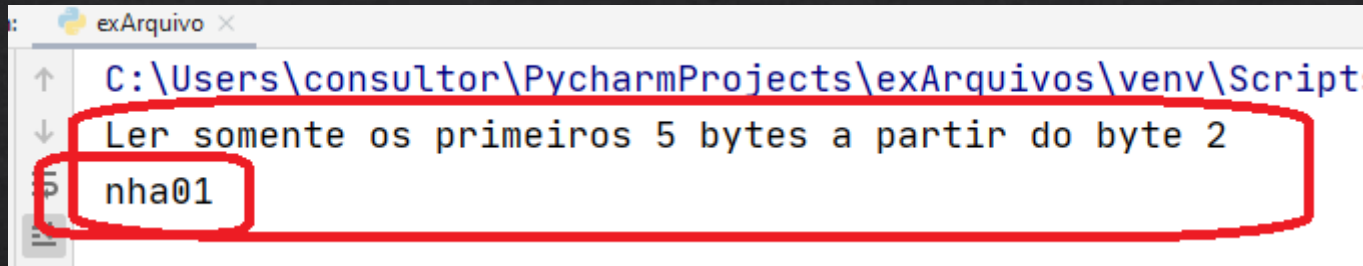
Carregando até um determinado byte:

O método “read”, irá carregar somente até o byte 5, mas antes posicionamos a partir do byte 2.

```
arquivo = open('C:\\Temp\\arq01.txt','r')  
  
def carregaAteTalByte():  
    arquivo.seek(2)  ##Posiciona no byte 2  
    print("Ler somente os primeiros 5 bytes a partir do byte %s" % arquivo.tell())  
    print(arquivo.read(5))  
    ##Le o arquivo somente até este byte  
    ##a partir da byte 2  
carregaAteTalByte()
```

# Arquivo – Exemplo - Leitura

O arquivo deverá ser apresentado desta maneira:



The screenshot shows a code editor window titled "exArquivo". The code is as follows:

```
C:\Users\consultor\PycharmProjects\exArquivos\venv\Script
Ler somente os primeiros 5 bytes a partir do byte 2
nha01
```

The code is enclosed in a red rectangular box. The first line is a file path, the second line is a comment, and the third line is a string.



# Arquivo – Exemplo - Leitura

Carregando até o fim do arquivo e atribuindo numa lista:

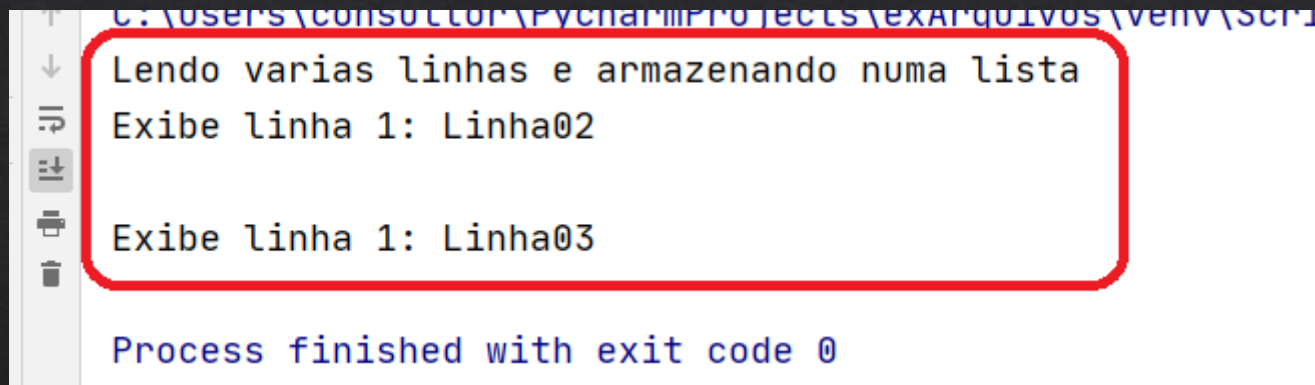
O “for”, irá percorrer todas as linhas do arquivo;

A “variável linha” receberá a instrução “readline” a cada interação que irá inserir no vetor.

```
def carregaArquivoAteoFim():  
    lista = []  
    print("Lendo varias linhas e armazenando numa lista")  
    for linha in arquivo:  
        lista.append(linha)  
    arquivo.close()  
    print("Exibe linha 2: %s" % lista[1])  
    print("Exibe linha 3: %s" % lista[2])  
    ##Caso aparecer o caractere de fim de linha use rstrip() para eliminar  
  
carregaArquivoAteoFim()
```

# Arquivo – Exemplo - Leitura

O arquivo deverá ser apresentado desta maneira:



A screenshot of a terminal window with a dark background. The window title bar shows the path 'C:\Users\consolator\PycharmProjects\exArquivos\venv\scri'. The terminal output is as follows:

```
Lendo varias linhas e armazenando numa lista  
Exibe linha 1: Linha02  
  
Exibe linha 1: Linha03  
  
Process finished with exit code 0
```

The first three lines of output are enclosed in a red rounded rectangle. On the left side of the terminal, there is a vertical toolbar with icons for running, debugging, and other actions.

# Arquivo

Você poderá realizar inúmeras combinações de comandos, mas basicamente leitura e escrita funcionam desta maneira.

# Dúvidas

???

# Referências

Docs python:

<https://docs.python.org/3/library/msilib.html?highlight=binary%20files#msilib.Binary>

Acesso em 01/08/2020.