

ARQUIVOS



ARQUIVOS

ARQUIVOS

ARQUIVOS

- Arquivos em programação Python é útil quando precisa armazenar dados permanentemente.
- Arquivos são uma excelente forma de entrada e saída de dados para programas. Com eles, podemos ler dados de outros programas e mesmo da internet.
- Arquivo é uma área em disco no qual pode ler e gravar informações. Essa área é gerenciada pelo sistema operacional do computador, ou seja, não há necessidade de se preocupar em como esse arquivo é organizado em disco.
- Do ponto de vista de programas, um arquivo é acessado por nome e é onde podemos ler e escrever linhas de texto ou dados em geral.
- Para acessar um arquivo, precisa-se abrir. Durante a abertura, deve ser informado o nome do arquivo, com o nome do diretório em que ele se encontra (se necessário) e que operações deseja realizar: leitura e/ou escrita

ARQUIVOS

ABERTURA DE ARQUIVOS

- Em Python, se abre um arquivo com a função **open**.
- A função **open** utiliza os parâmetros nome e modo.
- O **nome** é o nome do arquivo em si por exemplo leiname.txt.
- O **modo** indica as operações que deseja realizar:

MODO	OPERAÇÕES
r	Leitura
w	Escrita, apaga o conteúdo se já existir
a	Escrita, mas preserva o conteúdo se já existir
b	Modo binário
+	Atualização (leitura e escrita)

- Os modos podem ser combinados (“r+”, “w+”, “a+”, “r=b”, “w+b”, “a+b”).
- A função open retorna um objeto do tipo file (arquivo). É esse objeto que utiliza para ler e escrever os dados no arquivo.

ARQUIVOS

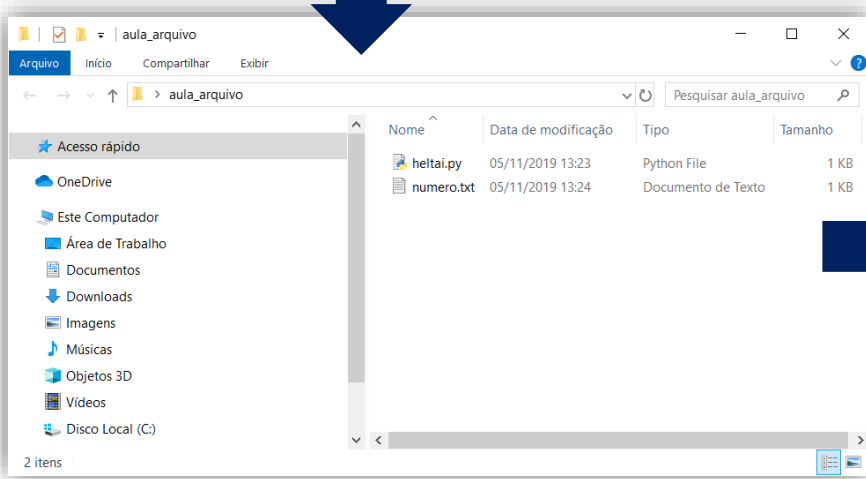
PROCESSO DE ESCRITA DE ARQUIVOS

- O método **write** é utilizado para escrever ou gravar dados no arquivo.
- **read** para ler e **close** para fechar.
- Ao trabalhar com arquivos, deve-se sempre realizar o seguinte ciclo:
 - ✓ Abertura
 - ✓ Leitura e/ou escrita
 - ✓ Fechamento
- A abertura realiza a ligação entre o programa e o espaço em disco.
- As etapas de leitura e/ou escrita são as operações que são realizadas pelo programa Python.
- O fechamento do arquivo é muito importante, pois cada arquivo aberto consome recursos do computador. Só o fechamento do arquivo garante a liberação desses recursos e preserva a integridade dos dados do arquivo.

ARQUIVOS

- **EXEMPLO:** Escrever em um arquivo numero.txt com 100 linhas e cada linha deve ser escrito um numero:

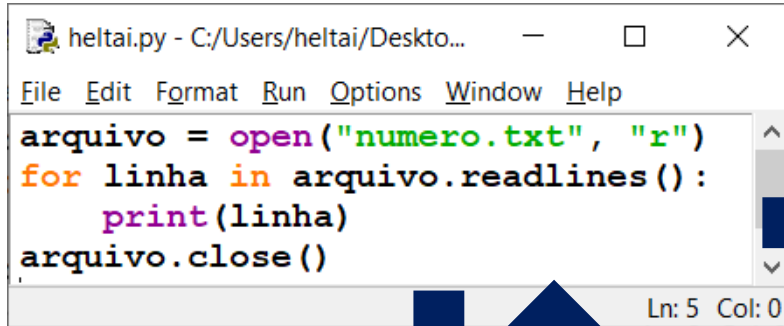
```
heltai.py - C:/Users/heltai/Deskto...  
File Edit Format Run Options Window Help  
arquivo = open("numero.txt", "w")  
for linha in range(1, 100):  
    arquivo.write(f"{linha}\n")  
arquivo.close()  
Ln: 6 Col: 0
```



```
numero.txt - Bloco de Notas  
Arquivo Editar Formatar Exibir Ajuda  
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
Ln 1, 100% Windows (CRLF) UTF-8
```

ARQUIVOS

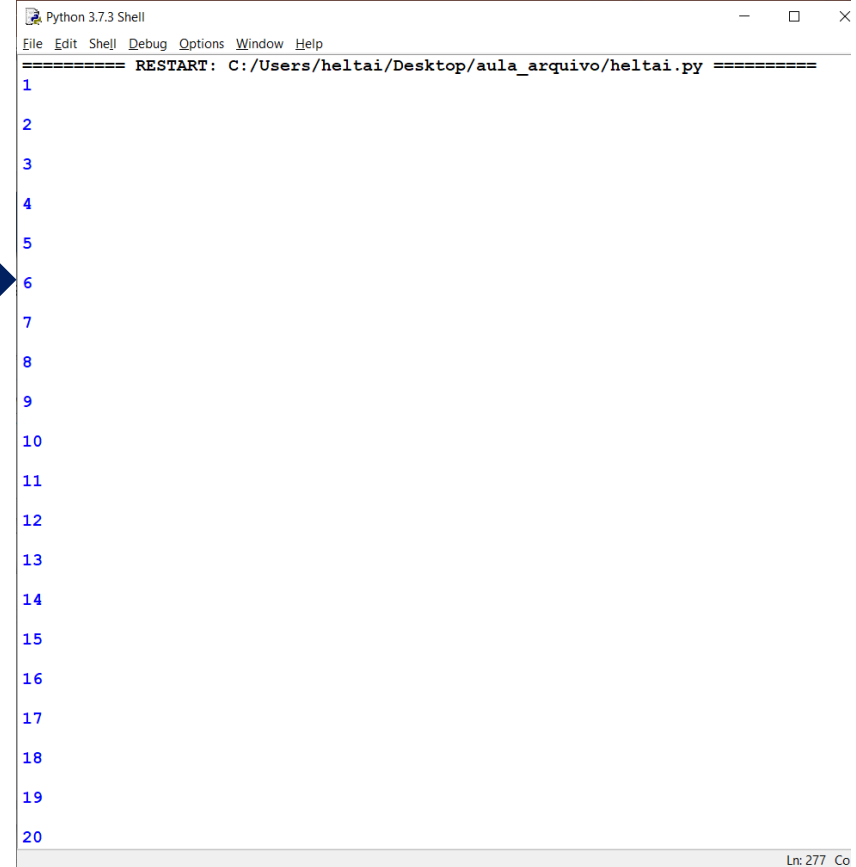
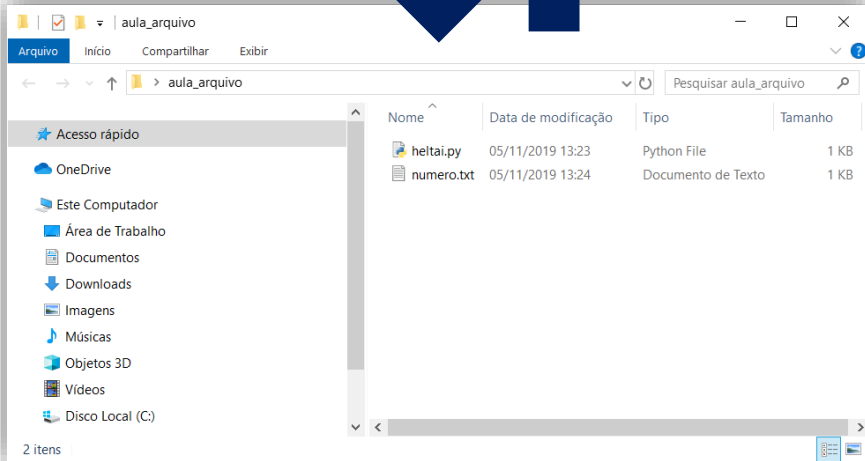
- **EXEMPLO:** Le o arquivo e imprimir suas linhas na tela:



heltai.py - C:/Users/heltai/Deskto...

```
File Edit Format Run Options Window Help
arquivo = open("numero.txt", "r")
for linha in arquivo.readlines():
    print(linha)
arquivo.close()
```

Ln: 5 Col: 0



Python 3.7.3 Shell

```
File Edit Shell Debug Options Window Help
===== RESTART: C:/Users/heltai/Desktop/aula_arquivo/heltai.py =====
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

Ln: 277 Co

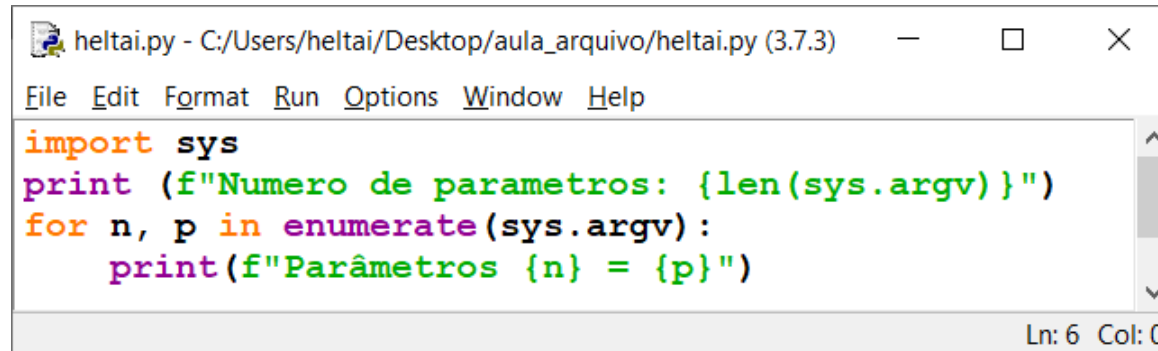


PARÂMETROS DA LINHA DE COMANDO

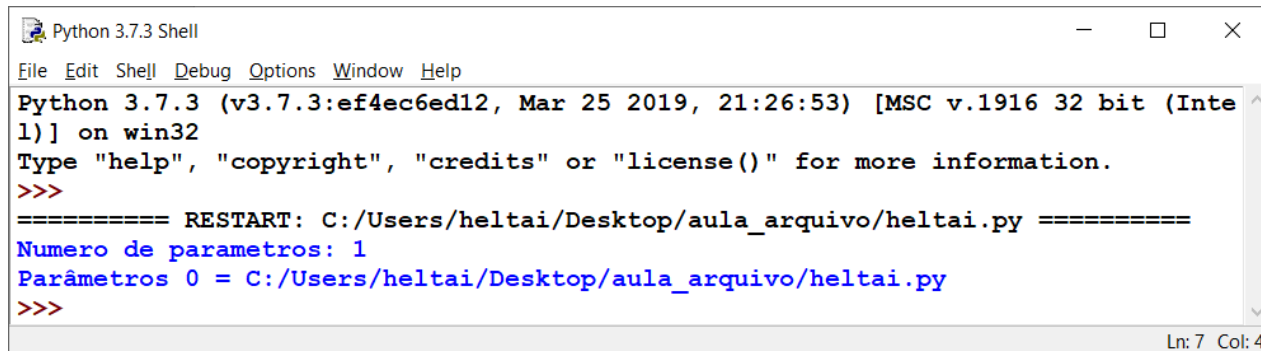
PARÂMETROS DA LINHA DE COMANDO

PARÂMETROS DA LINHA DE COMANDO:

- É possível acessar os parâmetros passados ao programa na linha de comando utilizando o módulo **sys** e trabalhando com a lista **argv**:



```
helta.py - C:/Users/helta/Deskto/aula_arquivo/helta.py (3.7.3)
File Edit Format Run Options Window Help
import sys
print (f"Numero de parametros: {len(sys.argv)}")
for n, p in enumerate(sys.argv):
    print(f"Parâmetros {n} = {p}")
Ln: 6 Col: 0
```



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/helta/Deskto/aula_arquivo/helta.py =====
Numero de parametros: 1
Parâmetros 0 = C:/Users/helta/Deskto/aula_arquivo/helta.py
>>>
Ln: 7 Col: 4
```



GERAÇÃO DE ARQUIVOS

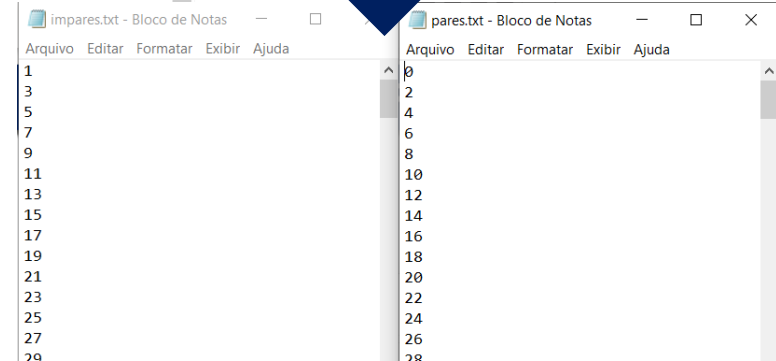
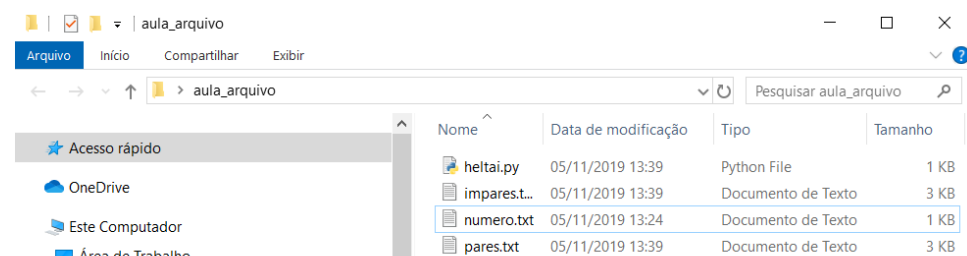
GERAÇÃO DE ARQUIVOS

GERAÇÃO DE ARQUIVOS:

- É possível gerar dois arquivos com 500 linhas cada. O programa distribui os números pares e ímpares em arquivos diferentes:

```
heltai.py - C:/Users/heltai/Desktop/aula_arquivo/heltai.py (3.7.3)
File Edit Format Run Options Window Help
with open("impares.txt", "w") as impares:
    with open("pares.txt", "w") as pares:
        for n in range(0, 1000):
            if n%2 ==0:
                pares.write(f"{n}\n")
            else:
                impares.write(f"{n}\n")
Ln: 9 Col: 16
```

```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 21:26:53) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/heltai/Desktop/aula_arquivo/heltai.py =====
>>> |
Ln: 5 Col: 4
```





LEITURA E ESCRITA DE ARQUIVOS

LEITURA E ESCRITA DE ARQUIVOS

LEITURA E ESCRITA DE ARQUIVOS:

- É possível realizar diversas operações com arquivos, entre elas ler, processar e gerar novos arquivos.
- Utilizando os arquivos anteriores, é possível filtrar de forma a gerar um novo arquivo, apenas com números múltiplos de 4:

The diagram illustrates the process of creating a new file from an existing one using Python. It shows a Python script in a Notepad window that reads lines from 'pares.txt' and writes only the numbers that are multiples of 4 to a new file 'multiplo_de_4.txt'. An arrow points to a File Explorer window showing the directory 'aula_arquivo' with files 'heltai.py', 'impares.txt', 'multiplo_de_4.txt', 'numero.txt', and 'pares.txt'. Another arrow points to a Notepad window showing the contents of 'multiplo_de_4.txt', which are numbers from 4 to 76 in increments of 4.

```
with open("multiplo_de_4.txt", "w") as multiplos4:
    with open("pares.txt") as pares:
        for l in pares.readlines():
            if int(l) % 4 == 0:
                multiplos4.write(l)
```

Nome	Data de modificação	Tipo	Tamanho
heltai.py	05/11/2019 13:45	Python File	1
impares.txt	05/11/2019 13:39	Documento de Texto	3
multiplo_de_4.txt	05/11/2019 13:45	Documento de Texto	2
numero.txt	05/11/2019 13:24	Documento de Texto	1
pares.txt	05/11/2019 13:39	Documento de Texto	3

Arquivo Editar Formatar Exibir Ajuda

4
8
12
16
20
24
28
32
36
40
44
48
52
56
60
64
68
72
76

Ln 1, 100% Windows (CRLF) UTF-8



PROF. VINICIUS HELTAI

vinicius.pacheco@docente.unip.br

www.heltai.com.br

(11) 98200-3932



Obrigado!

