



**INSTITUTO FEDERAL**

Catarinense

Campus Camboriú

## AULA 9 (Arquivos)

**Professora: Lidiane Visintin**

*lidiane.visintin@ifc.edu.br*

**Professor: Rafael de Moura Speroni**

*rafael.speroni@ifc.edu.br*

# Objetivo

---

- Compreender o conceito de **arquivos**.
  - Declaração
  - Abrindo arquivos
  - Funções para escrita e leitura

# Arquivos em Python



- Arquivos podem representar diversas coisas.
  - Será trabalhado apenas **arquivos em disco**.
- Manipulação de arquivos.

# Binário vs texto

<b>Tipo</b>	<b>Vantagens</b>	<b>Desvantagens</b>
Texto	- <i>Facilidade de leitura</i>	- <i>Maior gasto de memória</i> - <i>Maior gasto de tempo em buscas</i>
Binário	- <i>Menor gasto de memória</i> - <i>Menor gasto de tempo em buscas</i>	<i>Dificuldade de leitura</i>

# Manipulando arquivos

---

Função	Descrição
<code>open()</code>	- Abrir um arquivo.
<code>close()</code>	- Fechar um arquivo.
	- Declara uma variável para um arquivo. Ao chamar a função <code>open()</code> , ela cria uma instância da estrutura <code>FILE</code> e retorna um objeto para ela.

# Como abrir arquivos?

`File = open(<nome arquivo>, <modo>);`

modo de abertura do arquivo ( texto, binário, para leitura, escrita ou ambos).

nome do arquivo a ser aberto (com ou sem o caminho):

*Windows:*

```
char *arquivo = "c:\\dados\\lidiane\\info.txt";
```

*UNIX:*

```
char *arquivo = "c:/dados/\\/lidiane/info.txt";
```

# <modo>

<i><b>Modo</b></i>	<i><b>Significado</b></i>	<i><b>Se o arquivo NÃO existe...</b></i>	<i><b>Se o arquivo existe...</b></i>
r	Abre o arquivo para leitura.	O open() retorna <b>Erro.</b>	
w	Abre o arquivo para escrita.	O arquivo é criado.	O arquivo é <u>apagado</u> sem qualquer aviso e um novo arquivo vazio é criado em seu lugar.
a	Abre o arquivo para adicionar novos caracteres.	O arquivo é criado.	O arquivo é aberto para adição de caracteres no fim.

# <modo>

<i><b>Modo</b></i>	<i><b>Significado</b></i>	<i><b>Se o arquivo NÃO existe...</b></i>	<i><b>Se o arquivo existe...</b></i>
r+	Abre o arquivo para leitura e escrita.	O open() retorna <b>Erro.</b>	O arquivo é aberto para adição de caracteres no início, sobrescrevendo caracteres já existentes.
w+	Abre o arquivo para leitura e escrita.	O arquivo é criado.	O arquivo é aberto para adição de caracteres no início, sobrescrevendo o arquivo inteiro.
a+	Abre o arquivo para leitura e para adicionar novos caracteres.	O arquivo é criado.	O arquivo é aberto para adição de caracteres no fim.



# Abrindo um arquivo e escrevendo:

---

```
arquivo = open ("aula.txt", "w")  
  
for linha in range(1,101):  
    arquivo.write(f"{linha}\n")  
  
arquivo.close
```

# Abrindo um arquivo e lendo o seu conteúdo:

```
arquivo = open ("aula.txt", "r")  
  
for linha in arquivo.readlines():  
    print(linha)  
  
arquivo.close
```

# Usando with:

---

```
with open ("aula.txt", "r") as arquivo:  
    for linha in arquivo.readlines():  
        print(linha)
```

# Gerando arquivos:

---

```
#gravando números pares e impares em arquivos
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")
```

# Gerando arquivos:

```
#gravando numeros pares e impares em arquivos
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")
```

# Gerando arquivos:

```
#gravando numeros pares e impares em arquivos
with open ("impares.txt", "w") as impares, 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")
```

# Leitura e escrita

```
#lendo de um arquivo e gravando em outro arquivo
with open ("multiplos_de_4.txt", "w") as multiplos4:
    with open ("pares.txt", "r") as pares:
        for l in pares.readlines():
            if int(l) % 4 == 0:
                multiplos4.write(f"{l}\n")
```

# Processamento de um arquivo

---

Arquivo de entrada “entrada.txt”

;Esta linha não deve ser impressa

>Esta linha deve ser impressa a direita

\*Esta linha deve ser centralizada

Uma linha norma

Outra linha normal



# Processamento de um arquivo

```
#processamento de um arquivo
```

```
LARGURA = 100
```

```
with open("entrada.txt") as entrada:
    for linha in entrada.readlines():
        if linha[0] == ";":
            continue
        elif linha[0] == ">":
            print(linha[1:].rjust(LARGURA))
        elif linha[0] == "*":
            print(linha[1:].center(LARGURA))
        else:
            print(linha)
```

# Processamento de um arquivo

Arquivo de entrada “entrada.txt”

;Esta linha não deve ser impressa

>Esta linha deve ser impressa a direita

\*Esta linha deve ser centralizada

Uma linha norma

Outra linha normal

```
Esta linha deve ser impressa a direita
```

```
Esta linha deve ser centralizada
```

```
Uma linha norma
```

```
Outra linha normal
```

# Referências



## Referências Básicas

FORBELLONE, André Luiz Villar; EBERSPÄCHER, Henri Frederico. **Lógica de programação: a construção de algoritmos e estruturas de dados**. 3. ed. Pearson Prentice Hall. 2005

MANZANO, José Augusto N. G; OLIVEIRA, Jayr Figueiredo de. **Algoritmos: lógica para desenvolvimento de programação de computadores**. 27. ed.. Érica. 2014

## Referências Complementares

MENEZES, Nilo Ney de Coutinho. **Introdução a programação com Python**. 3ª Ed. Novatec. 2019.

CORMEN, Thomas H et al. **Algoritmos: teoria e prática**. 2. ed. Elsevier, Campus,. 2002

## Referências na Internet

<https://docs.python.org/3/>

<https://www.w3schools.com/python/default.asp>

<https://panda.ime.usp.br/pensepy/static/pensepy/10-Arquivos/files.html>