

DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON

Tema 2 - Aula 03 - Manipulação de dados.



Prof. Ronaldo Candido
ronaldo.candido@estacio.br



Estácio

2023.2

1

DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON

Objetivos da aula

- **Identificar as funções de manipulação de arquivos.**
- **Praticar o uso de funções de manipulação de arquivos em um cenário proposto.**


Conteúdo Programático

1. Funções de manipulação de arquivos.
2. Abertura, leitura, escrita e fechamento de arquivos.
3. Aplicação em Python.
4. Atividade verificadora de aprendizagem.
5. Aprenda+.

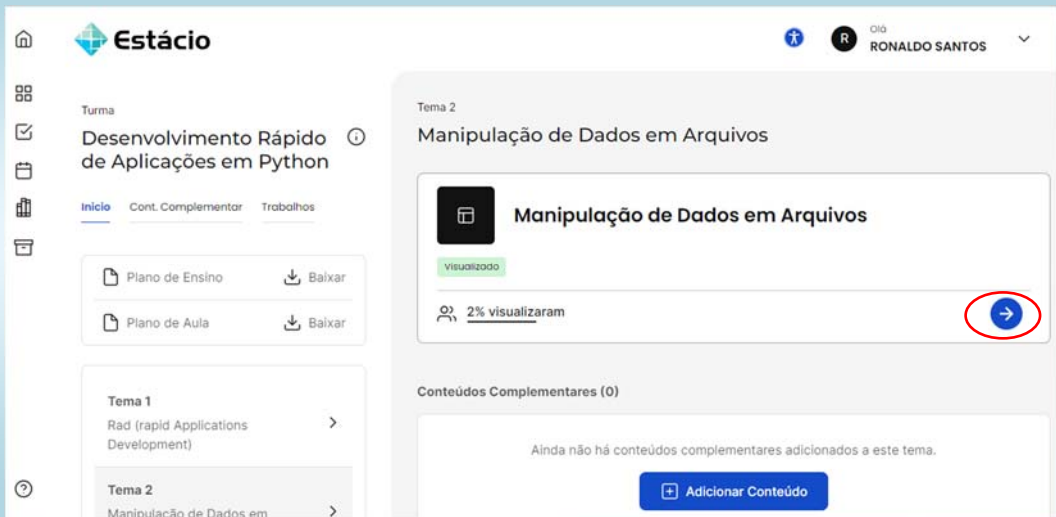
Prof. Ronaldo Candido

AULA 03

2

DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON 


Sala de Aula Virtual (SAVA)



Conteúdo digital


Prof. Ronaldo Candido AULA 03

3

DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON 

Situação-problema

- Suponha uma aplicação de registro de notas de alunos em uma instituição de ensino de pequeno porte. Um problema fundamental a ser tratado por essa aplicação é implementar alguma solução de persistência de dados.
- Como implementar essa aplicação com persistência de dados ?

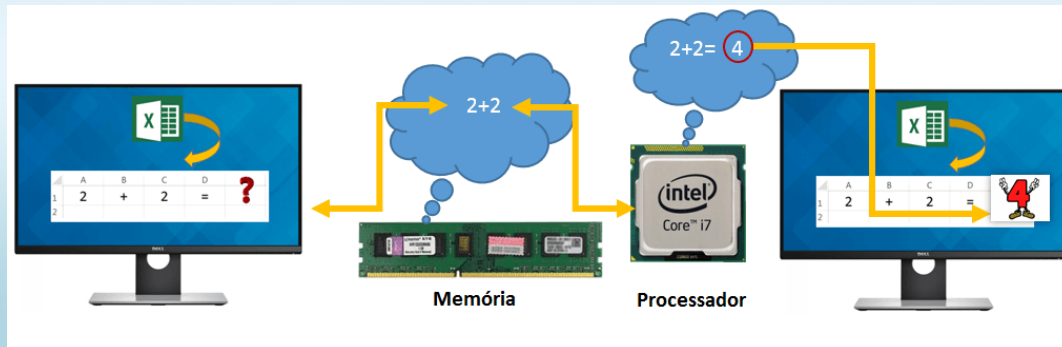


Prof. Ronaldo Candido AULA 03

4

Dados temporários

- Todos os dados trabalhados em nossos programas são armazenados na memória principal.
- A memória principal é volátil (RAM).
- Logo, ao terminar a execução do programa, os dados somem.



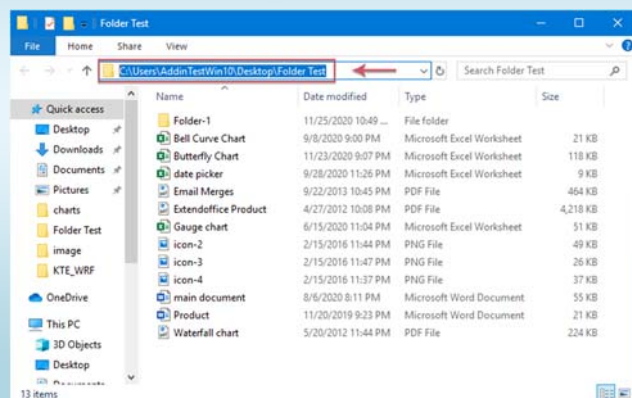
Prof. Ronaldo Candido

AULA 03

5

Arquivos

- Um arquivo é um registro armazenado em dispositivos de memória secundária.
- Armazenamento persistente.
- Disco rígido (HD), SSD, pen-drive, etc.
- O nome de um arquivo é composto por:
 - O diretório +
 - Seu nome e extensão.
 - Exemplo: `c:\documentos\consulta.txt`



Prof. Ronaldo Candido

AULA 03

6

Abrindo arquivos

- Para ler ou escrever um arquivo em um programa **Python** é preciso primeiro abri-lo:
SINTAXE: `arquivo = open(nomearquivo, modo)`
- A função **`open()`** recebe dois parâmetros:
 - Nome do arquivo (diretório + nome do arquivo + extensão)
 - Modo de acesso: `w` (write / escrita), ou `r` (read / leitura)
- A função **`open`** retorna uma instância do arquivo a ser manipulado.
- Caso não exista o arquivo, o mesmo será criado.

Caractere	O que faz
r	Abre para leitura (modo padrão)
w	Abre para escrita, apagando o conteúdo anterior.
x	Abre somente para criação e falha se o arquivo já existe.
a	Abre para escrita, adicionando novo conteúdo no arquivo.
b	Modo binário
t	Modo texto (padrão)
+	Abre um arquivo no disco para atualização (leitura e escrita)

Prof. Ronaldo Candido

AULA 03

7

Criando arquivos

- Para ler ou escrever um arquivo em um programa **Python** é preciso primeiro abri-lo:
#Simplesmente cria um arquivo vazio
`arq = open('teste.dat', 'w')`
`print (arq)`
`arq.close()`
`print("\nArquivo vazio criado com sucesso!")`

```
= RESTART: C:/2022_win10pro/estacio/2022.1/Desenv_Rápido_Python/aulas/zips/RAD_au
la03/criar_arquivo.py
<_io.TextIOWrapper name='teste.dat' mode='w' encoding='cp1252'>

Arquivo vazio criado com sucesso!
>>> |
```

Prof. Ronaldo Candido

AULA 03

8

Escrevendo arquivos

- O método `write()` permite escrever dados dentro do arquivo.
- A função recebe um parâmetro do tipo `string`.
- O arquivo foi aberto no modo `"w"` (escrita).

#função para gravar dados no arquivo

`def gerarListaOrdenada():`

`quantidade = int(input('Informe a quantidade de elementos:'))`

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

#gerar registro com os números

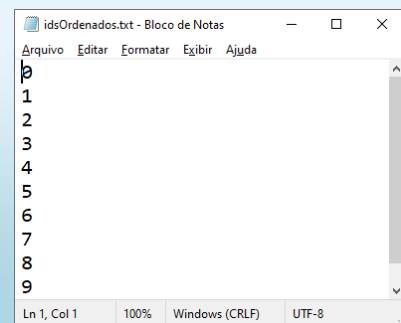
`for elemento in range(quantidade):`

`arquivo.write(str(elemento)+'\n')`

`arquivo.close()`

#executar a função

`gerarListaOrdenada()`



Escrevendo arquivos (continuação)

- O método `writelines()` permite escrever dados dentro do arquivo de uma só vez.
- A função recebe uma lista de `strings` como parâmetro.
- O arquivo foi aberto no modo `"w"` (escrita).

```
def gerarListaOrdenada():
    lista = []
    quantidade = int(input('Informe a quantidade de elementos: '))
    arquivo = open('idsOrdenados.txt', 'w')

    for elemento in range(quantidade):
        lista.append(str(elemento)+"\n")

    arquivo.writelines(lista)
    arquivo.close()
```


Lendo arquivos

- Uma vez que um arquivo foi aberto, podemos lê-lo:

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

- A função `read()` lê o conteúdo do arquivo:

- Retornará todo o conteúdo do arquivo.
- A função pode receber um parâmetro inteiro correspondente a quantidade de caracteres a ser lido.

```
= RESTART: C:/2022_win10pro/estacio/2022.1/Desenv_Rápido_Python/aulas/zips/RAD_a
ula03/ler_arquivo.py
0
1
2
3
4
5
6
7
8
9
```

Lendo arquivos de texto

- O método `readline()` lê todos os caracteres da próxima linha:

```
print("\n\t\t Abrindo arquivo desordenado.")

arquivo = open('idsOrdenados.txt', 'r')
linha = "."
while linha != "":
    linha = arquivo.readline()
    print(linha)

print("\n\t\t Fim do arquivo.")
arquivo.close()
```

- O método `readlines()` retorna todas as linhas do arquivo em uma lista :

```
arquivo = open('idsOrdenados.txt', 'r')
linhas = arquivo.readlines()
for linha in linhas:
    print(linha)

print("\n\t\t Fim do arquivo.")
arquivo.close()
```

Fechando arquivos

- A função `close()` informa que o programa terminou de usar o arquivo.
- Exemplo de cópia de conteúdo de arquivo :

```
def copiaArquivo(velhoArquivo, novoArquivo):
    f1 = open(velhoArquivo, "r")
    f2 = open(novoArquivo, "w")
    while 1:
        texto = f1.read(50)
        if texto == "":
            break
        f2.write(texto)
    f1.close()
    f2.close()
    return
```

Fechando arquivos (continuação)

- Exemplo de ordem de dados invertida :

```
def gerarListaInversamenteOrdenada():
    lista = []
    quantidade = int(input('Informe a quantidade de elementos: '))
    arquivo = open('idsOrdenadosInv.txt', 'w')

    quantidade -= 1
    while quantidade >= 0:
        lista.append(str(quantidade) + "\n")
        quantidade -= 1

    arquivo.writelines(lista)
    arquivo.close()
```

Atividade verificadora de aprendizagem

1. Desenvolva um programa em **Python** que escreva em disco um arquivo com números ordenados crescentemente de 1 a 100. Cada número deve ser separado por ";". O arquivo deve se chamar "crescente.txt".
2. Voltando ao cenário apresentado na situação-problema, que trata de um sistema de registro de notas de alunos em uma pequena instituição de ensino, desenvolver uma solução capaz de persistir (inserir) e ler os dados de notas de alunos em arquivos.
 - O programa deve registrar o nome, email e curso do aluno.
 - Cada novo registro deve ser armazenado em arquivo.
 - O usuário deve ter as seguintes opções:
 - ✓ Cadastrar um novo aluno.
 - ✓ Listar os alunos cadastrados.
 - ✓ Buscar um aluno pelo nome.

Aprenda +

- Assistir o vídeo: Canal DevPro. "Manipulando arquivos com Python". Disponível em: <<https://www.youtube.com/watch?v=utfmY8y6x50>>. Acesso em: 3 ago. 2023.

Atividade Autônoma Aura:

Questão 1) Existem dois modos de se trabalhar com arquivos, onde um é adequado para trabalhar com arquivos que contêm imagem, som e vídeo, dentre outros, o segundo modo pode ser aberto em qualquer editor básico e ser editado, esses modos são:

- a) Sequencial e Texto
- b) Binário e Texto
- c) Multimídia e Binário
- d) Texto e Stream
- e) Multimídia e Stream

Aprenda + (continuação)

Questão 2) O Python conta com recursos voltados à gravação e à leitura de arquivos, sejam eles binários ou texto. Nos arquivos do tipo texto a primeira providência é abrir o arquivo utilizando o método:


- a) `open("nome_arquivo",w)`
- b) `abrir("nome_arquivo",r)`
- c) `read("nome_arquivo",+wr)`
- d) `get("nome_arquivo",w)`
- e) `readline("nome_arquivo",r)`



Para a próxima aula...

- Leitura e resolução dos exercícios propostos no livro **"BANIN, S. L. Python 3 Conceitos e Aplicações. Uma Abordagem Didática"**. Capítulo 4: Tipos estruturados sequenciais em Python.
- Conteúdo digital da disciplina, Tema 2 - "Manipulação de Dados", Módulo 2: "Funções de manipulação de strings" e Módulo 2: "Exceções na manipulação de arquivos e outras operações".
- Estudar a Aula 04 de **DESENVOLVIMENTO RÁPIDO EM PYTHON** no SAVA previamente.




DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON  **Estácio**

Referências

BANIN, S. L. Python 3 Conceitos e Aplicações. Uma Abordagem Didática. 1a. ed. São Paulo: Érica, 2018. Disponível em:
<https://integrada.minhabiblioteca.com.br/#/books/9788536530253/>. Acesso em: 16 fev. 2023.

PERKOVIC, L. Introdução à Computação Usando Python - Um Foco no Desenvolvimento de Aplicações. 1a. ed. Rio de Janeiro: LTC, 2016. Disponível em:
<https://integrada.minhabiblioteca.com.br/reader/books/9788521630937/>. Acesso em: 16 fev. 2023.

SEBESTA, R. W. Conceitos de Linguagens de Programação. 11a. ed. Porto Alegre: Bookman, 2018. Disponível em:
<https://integrada.minhabiblioteca.com.br/#/books/9788582604694/>. Acesso em: 14 fev. 2023.



Prof. Ronaldo Candido AULA 03

19

DESENVOLVIMENTO RÁPIDO DE APLICAÇÕES EM PYTHON

Dúvidas, sugestões ou análises ???


 **Estácio**

Prof. Ronaldo Candido
ronaldo.candido@estacio.br

20