

Manipulação de Arquivos



Reflexão

- Trabalhar com dados é o foco de todas as aplicações
 - São nos dados que os usuários estão interessados
 - Consequentemente, armazenar dados de maneira permanente é importante!
- O dado na memória é temporário (dura apenas enquanto a máquina estiver funcionando)
- Dispositivos de armazenamento permanente permite manter os dados mesmo depois da máquina ser desligada
 - Mídias: disco rígido, Solid State Drive (SSD), pendrive...



Arquivos

- Dados são armazenados em Arquivos
 - Exemplo: em um editor de texto, os dados digitados estão sendo armazenados em arquivos
- Usualmente, arquivos possuem extensões (exe, jpg, py, txt...) que definem o tipo de dado armazenado neles
- Qualquer aplicação deve conhecer a estrutura do arquivo para poder manipular os dados armazenados
- Nessa disciplina, iremos trabalhar com **arquivos do tipo texto**



Arquivos Texto

- Um arquivo texto é um arquivo que contém caracteres imprimíveis e espaços, organizados dentro de linhas separadas por caracteres de nova linha (\n).
- Um arquivo texto não necessita possuir uma extensão específica, embora seja muito utilizada a extensão 'txt'.
- Arquivos fonte (por exemplo, '.py') são arquivos texto.
- Arquivos html, xml, csv também são exemplos de arquivos texto.

≡ Operações básicas em arquivos

- Devemos **abrir** (*open*) arquivos antes de usá-los e **fechar** (*close*) os arquivos depois de que tivermos terminado de utilizá-los.
 - **open** – Abre e retorna uma referência para o arquivo chamado.
 - `<ref_arquivo>=open (<nome_arquivo>,'r')` – para abrir no modo leitura 'r'
 - `<ref_arquivo>=open (<nome_arquivo>,'w')` – para abrir no modo escrita 'w'
'w' cria um arquivo `<nome_arquivo>` para gravação (deixando-o vazio mesmo que já exista)
 - `<ref_arquivo>=open (<nome_arquivo>,'a')` – para abrir no modo adição 'a'
'a' abre o arquivo `<nome_arquivo>` para que se possa gravar dados (não apaga o arquivo já existente). Entretanto, se `<nome_arquivo>` não existir, ele será criado
 - Os modos 'r','w' e 'a' podem ser combinados com '+' para abrir o arquivo no modo leitura e escrita
 - **close** – fecha o arquivo cuja referencia foi obtida pela instrução open.
 - `<ref_arquivo>.close()`

≡ Operações básicas em arquivos

- Após aberto o arquivo, podemos ler (*read*) os dados contidos nele ou escrever (*write*) novos dados.
 - **readline** – retorna uma linha de um arquivo.
 - `<ref_string>=<ref_arquivo>.readline()`
 - **readlines** – retorna uma lista com todas as linhas de um arquivo
 - `<ref_lista>=<ref_arquivo>.readlines()`
 - **read** – retorna todos os caracteres de um arquivo.
 - `<ref_string>=<ref_arquivo>.read()`
 - **write** – escreve uma string em um arquivo.
 - `<ref_arquivo>.write(<string>)`

≡ Referência (posição) no Arquivo

- Os arquivos são **sequências de bytes**.
- O primeiro byte está na posição 0, o segundo na posição 1, etc.
- Todo arquivo **guarda a posição/referência** do dado a ser lido ou gravado.
 - Quando **aberto** no modo **r** ou **w** esta referência está no **início do arquivo**
 - Quando **aberto** no modo **a** esta referência está no **final do arquivo**
 - A cada **read()** ou **readlines()** a referência se move para o **final do arquivo**
 - A cada **readline()** a referência se desloca para a **próxima linha**

≡ Referência (posição) no Arquivo

- ***seek*** – permite mudar a referência do dado a ser lido/gravado a partir de uma determinada posição
 - `<ref_arquivo>.seek(<deslocamento>,<origem>)`
deslocamento: número de bytes que a posição de referência será deslocada
origem: 0 – início do arquivo, 1 – posição atual, 2 – final do arquivo
- ***tell*** – informa a posição (em bytes) da referência do registro no arquivo
 - `<ref_arquivo>.tell()`

≡ Exemplo 1: criando um novo arquivo

- Abrindo um arquivo no modo gravação e escrevendo três linhas nele.

```
>>> arq = open('arquivo.txt', 'w')
>>> arq.write('Primeira linha\nSegunda linha\n')
29
>>> arq.write('Terceira linha\n')
15
>>> arq.close()
```

- O arquivo 'arquivo.txt' será criado no diretório corrente com o seguinte conteúdo:

```
Primeira linha
Segunda linha
Terceira linha
```

≡ Exemplo 2: adicionando dados a um arquivo existente

- Adicionando uma nova linha ao arquivo existente:

```
>>> arq = open('arquivo.txt', 'a')
>>> arq.write('Nova linha\n')
11
>>> arq.close()
```

- O arquivo 'arquivo.txt' passará a conter o seguinte conteúdo:

```
Primeira linha
Segunda linha
Terceira linha
Nova linha
```

≡ Exemplo 3: lendo um arquivo usando *read*

- Lendo todos os dados do arquivo:

```
>>> arq = open('arquivo.txt', 'r')
>>> texto = arq.read()
>>> texto
'Primeira linha\nSegunda linha\nTerceira linha\n'
>>> arq.close()
```

≡ Exemplo 4: lendo um arquivo usando *readline*

- Lendo as linhas do arquivo:

```
>>> arq = open('arquivo.txt', 'r')
>>> linha = arq.readline()
>>> linha
'Primeira linha\n'
>>> linha = arq.readline()
>>> linha
'Segunda linha\n'
>>> linha = arq.readline()
>>> linha
'Terceira linha\n'
>>> linha
''
>>> arq.close()
```

- obs: ***read*** retorna a string nula quando a referência está no final do arquivo.

≡ Exemplo 5: lendo um arquivo usando *readline* (com *while*)

- Lendo as linhas do arquivo (com o comando ***while***):

```
arq = open('arq.txt', 'r')
linha = arq.readline()
while linha:
    print(linha, end='')
    linha = arq.readline()
arq.close()
```

- Usando ***loop infinito*** e ***break***:

```
arq = open('arq.txt', 'r')
while True:
    linha = arq.readline()
    if linha == '':
        break
    print(linha, end='')
arq.close()
```

- Será exibido:

```
Primeira linha
Segunda linha
Terceira linha
```

≡ Exemplo 6: percorrendo um arquivo com o comando *for*

- O comando ***for*** pode ser utilizado para **iterar sobre cada linha** do arquivo

```
arq = open('arq.txt', 'r')
for linha in arq:
    print(linha, end='')
arq.close()
```

- Será exibido:

```
Primeira linha
Segunda linha
Terceira linha
```

- Obs: o uso do comando ***for*** para leitura em arquivos não é comum em outras linguagens de programação

≡ Exemplo 7: lendo um arquivo existente usando *readlines*

- Criando uma lista com as linhas do arquivo:

```
>>> arq = open('arquivo.txt', 'r')
>>> dados = arq.readlines()
>>> dados
['Primeira linha\n', 'Segunda linha\n', 'Terceira linha\n']
```

- A lista ***dados*** é criada com cada item contendo uma linha do arquivo.

≡ Exemplo 8: usando *seek()* e *tell()*

```
arq=open('cadastro.txt','a+')
arq.seek(0)
pos=arq.tell()
linha=arq.readline()
print('Posição - Dados no Arquivo\n')
while linha:
    print(pos, ' - ', linha)
    pos=arq.tell()
    linha=arq.readline()
print('\n\nLeitura direta do registro 17\n')
arq.seek(17)
pos=arq.tell()
linha=arq.readline()
print(pos, ' - ', linha)
arq.close()
```



Posição - Dados no Arquivo

0 - Ana,22,feminino

17 - Pedro,21,masculino

37 - Rui,18,masculino

Leitura direta do registro 17

17 - Pedro,21,masculino