Módulo I – Fundamentos Computacionais
Ses 3 - Aula 2 - Entrada e Saída em Arquivos

Prof. Dr. Luiz Alberto Vieira Dias
Prof. Dr. Lineu Mialaret

Arquivos em Python

- Arquivo representa uma localização nomeada em disco para armazenar dados, sendo utilizado para realizar o armazenamento de forma não volátil (permanente)
  - Quando se lê ou se grava dados em arquivo, é necessário abrí-lo primeiro
  - Uma vez aberto, realizam-se as manipulações de dados
  - Finalizadas as manipulações, os recursos alocados com o arquivo devem ser liberados com o fechamento do arquivo
- Desta forma, no Python uma manipulação de dados em arquivo ocorre da seguinte forma:
  - Abrir o arquivo
  - Ler ou gravar dados
  - Fechar o arquivo

Arquivos em Python (cont)

- No Python, os arquivos são acessados começando-se pela utilização da função pré-construída open(), que retorna um objeto (*proxy* ou *handler*) para interações com o arquivo
- Exemplo:

```
fp = open( 'sample.txt' ) # open file in current directory
```

  - a instrução open() acima tenta abrir um arquivo chamado sample.txt no diretório corrente, retornado um proxy que permite acesso de leitura ao arquivo
- A função open () aceita um segundo argumento opcional que determina o modo de acesso ao arquivo. Os modos são:
  - r para leitura
  - w para escrever
  - a para anexar ao final de um arquivo existente

Arquivos em Python (cont.)

- O padrão para leitura é em modo texto. Há também o modo binário de leitura (arquivo exe ou de imagens)

```
f = open("test.txt")      # equivalent to 'r' or 'rt'
f = open("test.txt",'w')  # write in text mode
f = open("img.bmp",'r+b') # read and write in binary mode
f = open("test.txt",mode = 'r',encoding = 'utf-8') # using utf8 enconding
```

Mode	Description
'r'	Open a file for reading. (default)
'w'	Open a file for writing. Creates a new file if it does not exist or truncates the file if it exists.
'x'	Open a file for exclusive creation. If the file already exists, the operation fails.
'a'	Open for appending at the end of the file without truncating it. Creates a new file if it does not exist.
't'	Open in text mode. (default)
'b'	Open in binary mode.
'+'	Open a file for updating (reading and writing)

Arquivos em Python (cont.)

- Quando se processa um arquivo, o *proxy* mantém a posição corrente dentro do arquivo como a distância (*offset*) do começo do arquivo, medida em bytes
  - Quando se abre um arquivo com o modo `r` ou `w`, a posição inicialmente é 0
  - Quando se abre com o modo `a`, a posição incialmente é no final do arquivo

Leitura de Arquivos em Python

- Via um proxy
- read()

  Retorna uma string contendo os próximos k bytes do arquivo, a partir da posição corrente

  - Sem parâmetros, retorna o conteúdo do arquivo

test.txt

```
This is my first file
This file
contains three lines
```

```
>>> f = open("test.txt",'r',encoding = 'utf-8')
>>> f.read(4)    # read the first 4 data
'This'

>>> f.read(4)    # read the next 4 data
' is '

>>> f.read()     # read in the rest till end of
file
'my first file\nThis file\ncontains three
lines\n'

>>> f.read()  # further reading returns empty
sting
''
```

Leitura de Arquivos em Python (cont.)

- Pode-se mudar a posição corrente do cursor no arquivo usando-se o método seek()
- O método tell() retorna a posição corrente do cursor no arquivo (em bytes)

```
>>> f.tell()    # get the current file position
56

>>> f.seek(0)   # bring file cursor to initial
position
0

>>> print(f.read())  # read the entire file
This is my first file
This file
contains three lines
```

Leitura de Arquivos em Python (cont.)

- Pode-se ler um arquivo linha a linha usando-se um laço for (for loop)

```
>>> for line in f:
...     print(line, end = '')
...
This is my first file
This file
contains three lines
```

- Pode-se usar também o método readline() para se ler linhas do arquivo. O método lê o arquivo até o caractere de nova linha.

```
>>> f.readline()
'This is my first file\n'
>>> f.readline()
'This file\n'
>>> f.readline()
'contains three lines\n'
>>> f.readline()
''
```

Leitura de Arquivos em Python (cont.)

- O método readlines() retorna uma lista das linhas remanescentes do arquivo inteiro

```
>>> f.readlines()
['This is my first file\n', 'This file\n', 'contains three lines\n']
```

Gravação em Arquivos em Python

- Para se gravar dados em arquivos, eles devem ser abertos com os modos ‘w’ (gravação), ‘a’ (anexação) ou ‘x’ (criação exclusiva)
- Arquivo aberto como modo ‘w’ sobrescreve arquivos existentes como o mesmo nome
- A gravação no arquivo é realizada utilizando-se o método write(), que retorna o número de bytes gravados no arquivo

```
>>> f = open("d:/test.txt",'w')
>>> f.write("my first file\n")
14
>>> f.write("This file\n\n")
11
>>> f.write("contains three lines\n")
21
```

Sintetizando Leitura e Gravação de Arquivos em Python

Calling Syntax	Description
fp.read( )	Return the (remaining) contents of a readable file as a string.
fp.read(k)	Return the next *k* bytes of a readable file as a string.
fp.readline( )	Return (remainder of) the current line of a readable file as a string.
fp.readlines( )	Return all (remaining) lines of a readable file as a list of strings.
for line **in** fp:	Iterate all (remaining) lines of a readable file.
fp.seek(k)	Change the current position to be at the k^{th} byte of the file.
fp.tell( )	Return the current position, measured as byte-offset from the start.
fp.write(string)	Write given string at current position of the writable file.
fp.writelines(seq)	Write each of the strings of the given sequence at the current position of the writable file. This command does *not* insert any newlines, beyond those that are embedded in the strings.
print(..., file=fp)	Redirect output of print function to the file.

Behaviors for interacting with a text file via a file proxy (named fp).

Fechamento de Arquivos em Python

- Quando se encerram as manipulações de dados em arquivos, eles devem ser fechados adequadamente
- O fechamento do arquivo libera recursos alocados ao arquivo e isto é feito utilizando-se o método close()
- O Pyhton tem mecanismo de coleta de lixo (*garbage collection*) que podem fazer a liberação destes recursos, mas é mais interessante o desenvolvedor faze isto

```
f = open("test.txt",encoding = 'utf-8')
# perform file operations
f.close()
```

Fechamento de Arquivos em Python (cont.)

- Um modo de se garantir o fechamento de um arquivo sem preocupação é abrir o arquivo utilizando-se a instrução with
- Isto garante que o arquivo será fechado quando o bloco de instruções dentro da cláusula with tiver terminado
- O desenvolvedor não precisa chamar explicitamente o método close() para fechar o arquivo

```
f = open("test.txt",encoding = 'utf-8')
# perform file operations
f.close()
```

```
with open("test.txt",encoding = 'utf-8') as f:
   # perform file operations
```

```
with open("test.txt",'w',encoding = 'utf-8') as f:
   f.write("my first file\n")
   f.write("This file\n\n")
   f.write("contains three lines\n")
```