MÉTODOS COMPUTACIONAIS

DR. MARCOS NAPOLEÃO RABELO
DR. WANDERLEI M. PEREIRA JUNIOR

Manipulação de arquivos

Grupo de Pesquisa e Estudos em Engenharia (GPEE)

MANIPULAÇÃO DE ARQUIVOS

É comum em computação científica a utilização de arquivos para manipulação de dados. Mas qual tipo de arquivo? A resposta é qualquer um. No nosso curso por exemplo utilizaremos em aplicações mais avançadas o emprego das informações contidas em um arquivo .txt, ou seja texto.

Em geral as operações sob arquivos são:

- Abertura;
- Operação que manipula os dados;
- Fechamento.

A manipulação de arquivos imprime uma série de facilidades no ato de se construir um algoritmo. Vamos utilizar alguns exemplos atuais: O tratamento de informações de um banco de dados que salva os ingredientes da dosagem de um concreto ou por exemplo o endereço e características de edificações. Como seria complexo armazenar uma matriz com todas essas informações?!

Arquivos como o .csv (Valores Separados por Vírgula ou em inglês *Comma-Separated Values*) são extensões de arquivo com grande importância na ciência de dados.

Claro que são diversos as possibilidades de operações com arquivos de dados, são exemplos bastante usuais: Planilhas eletrônicas, .csv e o .txt.

Em Python são diversos os recursos para manipulação de arquivos. Abaixo vamos exemplificar o comando `.open()` que é empregado para abrir um arquivo externo ao algoritmo, seja para proceder com uma leitura ou escrita por exemplo [1].

```
>>> arquivo = open("contatos.txt", "a")
```

Nessa linha executamos a abertura do arquivo `"contatos.txt"` e procedemos com o comando escrita ao final do arquivo. Fato que é permitido pela inserção da tag `"a"`.

Para então escrever ao final do arquivo utilizamos o comando `.write()` conforme a seguinte sintaxe

```
>>> arquivo.write("Olá, mundo!")
```

Existem diversas opções para leitura do conteúdo de um arquivo, por exemplo em Python são funções passíveis de utilização:

- `.read()` – Lê uma quantidade de bytes especificada;
- `.readline()` – Retorna uma *string* baseada na leitura da primeira linha do arquivo;
- `.readlines()` – Retorna uma lista de *strings* baseada na leitura do arquivo.

No Python o processo de escrita também pode ser realizado de forma similar ao `.read()`, porém com a utilização da função `.write()`. É muito comum em Python a escrita do texto em uma variável única utilizando o conceito de lista e depois executar o comando `.writelines()`[1].

```
>>> arquivo = open("texto.txt", "a")
>>> frases = []
>>> frases.append("TreinaWeb \n")
>>> frases.append("Python \n")
>>> frases.append("Arquivos \n")
>>> frases.append("Django \n")
>>> arquivo.writelines(frases)
```

Em caso de necessidade impressão nos arquivos de valores numéricos é comum utilizar a função *.format()*. Um exemplo dessa função pode ser visto a seguir:

```
>>> arquivo = open("texto.txt", "a")
>>> valores = []
>>> n1 = 2.5; n2 = 5.5; n3 = 24.89
>>> posproc.append('{:10.5f} {:10.5f} 0.0 {:10.5f}\n'.format(n1, n2, n3))
```

Basicamente o elemento `.format()` troca o elemento entre chaves pelo argumento informado na função, lembrando que o argumento entre chaves pode ser um padrão `:[tamanho].[precisão][tipo variável]`.

No caso do exemplo foi escrita uma variável com 10 espaços, 5 casas de precisão do tipo *float*.

Além da questão da manipulação do arquivo é importante sempre ao final do processo, de uso de um arquivo, fechar o mesmo. Para isso o Python e também outras linguagens se utilizam da função close. No caso do Python está função possui a seguinte sintaxe `.close()`.

REFERÊNCIAS

[1] Manipulando arquivos com Python. Blog da TreinaWeb 2020. https://www.treinaweb.com.br/blog/manipulando-arquivos-com-python/ (accessed May 19, 2021).