Exercícios do Curso de Python

Ficha 4

Ficheiros e tratamento de excepções

1. Suponha que executava os seguintes programas. Qual seria o resultado?

1.
```
out_file = open(“outFile.txt”, 'w')
for i in range(100):
    out_file.write(str(i))
```

2.
```
in_file = open(“inFile.txt”,'r')
indata = in_file.read()
in_file.close()
indata = in_file.read()
in_file.close()
```

3.
```
in_file = open(“inFile.txt”,'r')
print(in_file.readline())
in_file = open(“inFile.txt” ,'r'))
print(in_file.readline())
in_file.close()
```

2. Escreva um programa que leia um ficheiro de texto linha a linha e escreva o seu conteúdo no ecrã.

3. Modifique o programa da alínea anterior de forma a aparecer também o número de cada linha.

4. Dados referentes a observações são frequentemente guardados em ficheiros de texto. Por exemplo, as temperaturas lidas a várias horas do dia, ao longo de vários dias, podem ser guardadas num ficheiro de números em vírgula flutuante, onde cada linha contém os valores das várias temperaturas medidas num dia.

```
5.6 7.8 11.7 12.6 9.3 7.3
6.7 8.5 11.6 11.6 9.4 7.0
5.4 7.2 10.5 11.1 10.0 8.3
```

Utilizando a função media, escreva uma função médias que, dado o nome de um ficheiro de texto como o acima, imprima as temperaturas médias diárias. Deverá imprimir um valor por linha e tantos valores quantas as linhas do ficheiro. Sugestão: utilize o método string.split(s) para obter a lista de palavras existentes numa string. A função media deve apanhar a exceções referentes à utilização do ficheiro. Para isso utilize um **try: finally:** ou um **with**, para se assegurar que fecha o ficheiro. Exceções sobre a abertura do ficheiro deverão ser tratadas pelo chamador (ver exercício seguinte).

5. Escreva uma função principal sem parâmetros que pede ao utilizador o nome dum ficheiro de temperaturas, e chama a função medias passando o nome do ficheiro. Caso ocorra algum problema de acesso ao ficheiro, a função principal deve escrever Erro de I/O ao ler o ficheiro

6. Informação referente a símbolos químicos pode ser guardada em ficheiros de texto. Neste caso estamos interessados apenas no nome, número atómico e densidade (em g/dm3). Eis um exemplo:

```
Hélio 2 0.1786
Néon 10 0.9002
Argon 18 1.7818
Cripton 36 3.708
Xenon 54 5.851
Radônio 86 9.97
```

Hélio 2 0.1786 Néon 10 0.9002 Argon 18 1.7818 Cripton 36 3.708 Xenon 54 5.851 Radônio 86 9.97 2Escreva uma função linha_para_elemento que dada uma linha de um ficheiro de elementos (uma string), produz um dicionário com chaves 'nome', 'atomico' e 'densidade'. O nome é do tipo string, o número atómico do tipo int e a densidade do tipo float.

7. Utilizando a função elemento_para_string, escreva uma função escrever_elementos que, dada uma lista de dicionários representando elementos químicos e o nome de um ficheiro, escreve o conteúdo da lista no ficheiro.

8. O arranjo de átomos em moléculas pode ser descrito em formato textual, e portanto guardado em ficheiros. Cada molécula é descrita por um número variável de linhas: a primeira contém o nome da molécula, as subsequentes contêm informação sobre o átomo. A última linha, END, fecha a molécula. Cada átomo é composto pelo seu identificador, símbolo químico, e as coordenadas tri-dimensionais. Eis o exemplo de um ficheiro contendo duas moléculas.

```
COMPND          AMMONIA ATOM
ATOM            1 N 0.257 -0.363  0.000
ATOM            2 H 0.257  0.727  0.000
ATOM            3 H 0.771 -0.727  0.890
ATOM            4 H 0.771 -0.727 -0.890
END
COMPND        METHANOL
ATOM          1 C -0.748 -0.015  0.024
ATOM          2 O  0.558  0.420 -0.278
ATOM          3 H -1.293 -0.202 -0.901
ATOM          4 H -1.263  0.754  0.600
ATOM          5 H -0.699 -0.934  0.609
ATOM          6 H  0.716  1.404  0.137
END
```

Escreva uma função linha_para_atomo que, dada uma string contendo uma linha ATOM, devolve um dicionário com chaves 'símbolo', 'id', 'x', 'y', e 'z'.

Manipulação de CSV

1. Comma Separated Values (CSV) é o formato mais comum de importação/exportação de dados para folhas de cálculo e para bases de dados. A linguagem Python conta com um módulo para o efeito, CSV File Reading and Writing. O padrão mais comum de utilização do CSV para leitura é o seguinte:

```
import csv

with open('ficheiro.csv', 'rU') as ficheiro_csv:
leitor = csv.reader(ficheiro_csv, delimiter=';')
```

No código acima leitor é um iterador. Pode ser usado num ciclo for, como por exemplo:

```
for linha in leitor:
    <fazer algo com linha>
```

Alternativamente, podemos chamar o método leitor.next() para obter os sucessivos elementos no iterador. A exceção StopIteration é levandada quando tentamos fazer um **next()** e não há mais elementos no iterador. A razão para usar 'rU' está relacionado com a função **open()**, e não propriamente com CSV. O parâmetro 'U' (de Universal), aceita linhas com a convenção Unix ('\n') e Windows ('\r\n'). O delimitador ';' é importante pois o formato CVS para a língua portuguesa utiliza o ponto-e-vírgula como separador, uma vez que a vírgula é utilizada como separador da parte decimal dos números. Escreva uma função csvLinhaParaGrafico que leia dum ficheiro CSV o gráfico de uma função e o devolva. A função recebe o nome de um ficheiro. Assuma que o ficheiro é composto por duas linhas, a primeira contendo os valores das abcissas, a segunda os valores das ordenadas. Não se esqueça de converter as strings lidas do ficheiro em números em vírgula flutuante.

2. O padrão típico para escrever num ficheiro CSV é o seguinte.

```
import csv

with open('ficheiro.csv', 'wb') as ficheiro_csv:
    escritor = csv.writer(ficheiro_csv, delimiter=';')
    escritor.writerow([0, 1, 2, 3, 4, 5])
    escritor.writerow([0, 2, 4, 6, 8, 10])
```

   Escreva uma função graficoParaCsvLinha que receba o nome de um ficheiro e um gráfico (um par de listas), e escreva o gráfico no ficheiro do seguinte modo: a primeira linha contém as abcissas, a segunda as ordenadas

3. Escreva uma função graficosParaCsv que, dado o nome de um ficheiro e uma lista de gráficos (uma lista de listas de números), escreva os gráficos num ficheiro CSV. Organize a informação no ficheiro do seguinte modo: a primeira coluna contém as abcissas, a segunda coluna contém as ordenadas da primeira função, a terceira coluna contém as ordenadas da segunda função, e por aí em diante. Assuma que todas as funções têm listas de abcissas iguais