Programação II

Exercícios 1
Exceções e tratamento de ficheiros (revisão)

Universidade de Lisboa
Faculdade de Ciências
Departamento de Informática
Licenciatura em Tecnologias da Informação

2021/2022

1. Identifique o resultado de cada um dos seguintes excertos de código asumindo que o conteúdo do ficheiro `frutas.txt` é:

   ```
   kiwi
   pera
   maça
   ananás
   ```

   (a)
   ```
   in_file = open("frutas.txt",'r')
   indata = in_file.read()
   print(indata)
   ```

   (b)
   ```
   in_file = open("frutas.txt",'r')
   indata = in_file.readline()
   print(indata)
   ```

   (c)
   ```
   in_file = open("frutas.txt",'r')
   indata = in_file.readlines()
   print(indata)
   ```

   (d)
   ```
   in_file = open("frutas.txt",'r')
   indata = list(in_file)
   print(indata)
   ```

   (e)
   ```
   in_file = open("frutas.txt",'r')
   for i in range(5):
     print(in_file.readline())
   ```

2. Escreva uma função `le_ficheiro` que leia um ficheiro de texto linha a linha e escreva o seu conteúdo no ecrã.

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

4. Escreva uma função `escreve_ficheiro` que peça ao utilizador várias linhas e as escreva num ficheiro. O programa termina quando o utilizador introduzir uma linha vazia.

5. Escreva uma função `conta_linhas` que dado um nome de ficheiro, abra este ficheiro e conte o número de linhas de texto que contém.

6. Escreva uma função `conta_linhas_com_string` que dado um nome de ficheiro e uma string, conte o número de linhas que contém essa string.

7. Escreva uma função `conta_linhas_caracteres` que dado um nome de ficheiro, devolva um par com o número de linhas e o número de caracteres.

8. Escreva uma função `copia_ficheiro` que dados dois nomes de ficheiros, faz uma cópia dos conteúdos de um ficheiro para o outro.

9. Escreva uma função `copia_ficheiro_maiusculas` que dados dois nomes de ficheiros, faz uma cópia dos conteúdos de um ficheiro para o outro, convertendo todas as letras em maiúsculas. Utilize o método `.upper()`.

10. Escreva uma função `lista_floats` que dada uma lista de strings, cada uma representando um número, devolve uma outra lista com os números em vírgula flutuante correspondentes. Verifique que a função levanta a exceção `ValueError` quando pelo menos um elemento da lista não for convertível para **`float`**. Exemplos:

```
>>> lista_floats(['3.14', '1', '-0.4'])
[3.14, 1.0, -0.4]
>>> lista_floats(['3.14', 'um', '-0.4'])
...
ValueError: could not convert string to float: 'um'
```

11. Escreva uma função `media` que calcula a média de uma lista de números. Verifique que a função levanta a exceção `ZeroDivisionError` quando a lista estiver vazia.

```
>>> l = [1, 2.0]; media(l)
1.5
>>> l = []; media(l)
...
ZeroDivisionError: integer division or modulo by zero
```

12. 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 as funções `lista_floats` e `media` dos exercícios anteriores, escreva uma função `imprime_medias` 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 `imprime_medias` deve apanhar as exceções lançadas pelas funções `lista_floats` e `media`. No caso de divisão por zero deverá imprimir `"linha␣vazia"`; no caso de uma linha que contenha uma palavra que não representa um número em vírgula flutuante deverá imprimir `"linha␣mal␣formada"`.

    Uma das pré-condições da função `imprime_medias` é que o nome do ficheiro argumento representa um ficheiro válido, um ficheiro que pode ser aberto para leitura sem levantar exceção alguma.

    (a) Escreva a função `imprime_medias` que imprima algo (a média ou uma mensagem de erro) *para cada linha no ficheiro*. Utilize o comando `with`.

    (b) Adapte a versão da alínea anterior de modo a que a função pare ao primeiro erro, depois de escrever a mensagem de erro adequada.

13. Escreva uma função `principal` sem parâmetros que pede ao utilizador o nome de um ficheiro de temperaturas, e chama a função `imprime_medias` passando o nome do ficheiro. Apanhe a exceção `IOError` da função `imprime_medias` relativa à tentativa falhada de abrir o ficheiro e imprima uma mensagem adequada. Utilize a função **`input`** para ler uma linha do canal de entrada. Ignore exceção `EOFError` que esta função pode eventualmente levantar.

14. Por vezes os ficheiros de observações trazem informação sobre os dados na forma de linhas comentadas, cada qual iniciada pelo carater cardinal `#`. Eis um exemplo:

    ```
    # Localização: Observatório de Muge
    # 14/2/2016
    5.6 7.8 11.7 12.6 9.3 7.3
    ```

```
# 15/2/2016
6.7 8.5 11.6 11.6 9.4 7.0
# 16/2/2016
5.4 7.2 10.5 11.1 10.0 8.3
```

Escreva uma função `salta_comentario` que, dado um ficheiro aberto para leitura, devolva a primeira linha que não está comentada.

15. Usando a função `salta_comentario`, altere a função `imprime_medias` do exercício 12, de modo a ignorar linhas comentadas no ficheiro. Apelide-a de `medias_salta_comentario`.

16. Escreva uma função `lista_para_ficheiro` que, dada uma lista e o nome de um ficheiro, escreve os vários elementos da lista, um por linha, no ficheiro.

17. Usando a função `lista_para_ficheiro`, altere a função do exercício 12, de modo a escrever as médias num dado ficheiro. Apelide-a de `medias_para_ficheiro`. Deve receber duas strings com os nomes dos dois ficheiros envolvidos.