# Programação II

# Exercícios 8
# Programação de Sistema

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

2020/2021

1. Escreva um *script* `para_minusculas` que imprima no ecrã o conteúdo do ficheiro com todas as letras convertidas para minúsculas. Exemplo:

   ```
   $ cat mensagem.txt
   A Europa jaz, posta nos cotovellos:
   De Oriente a Occidente jaz, fitando,
   $ python para_minusculas.py mensagem.txt
   a europa jaz, posta nos cotovellos:
   de oriente a occidente jaz, fitando,
   ```

   Sugestão: proceda em dois passos:

   (a) Escreva uma funcão `para_minusculas(nome_ficheiro)` que imprime no ecrã o conteúdo de um ficheiro com todas as letras convertidas para minúsculas.

   (b) Baseado na função anterior, escreva agora o *script* `para_minusculas.py` que lê da linha de comandos o nome do ficheiro e imprime no ecrã o conteúdo deste com todas as letras convertidas para minúsculas.

2. `more` é um filtro para visualizar o conteúdo de um ficheiro, uma página de cada vez. Na mais simples utilização, `more` é lançado da linha de comandos deste modo:

   ```
   $ python more.py o_meu_ficheiro
   ```

   Neste caso a primeira página do ficheiro aparece no ecrã. Depois, sempre que o utilizador carregar em `Enter` aparecerá uma nova página. Qualquer outra tecla seguida de `Enter` termina o programa.

(a) Implemente a versão simplificada do *script* `more.py` indicada acima.

(b) Um exemplo de interação mais complexo especifica o tamanho de cada página (`-c`):

```
$ python more.py -c 20 more.py
```

O valor por omissão para o comprimento da página é 10. Implemente esta funcionalidade no seu *script*. Sugestão: pode utilizar `len(sys.argv)` para obter o número de argumentos.

(c) Utilizando o módulo `argparse`, estenda o seu *script* de modo a possuir as seguintes opções:

- `-h, --help` mostra como utilizar o programa;
- `-c N, --comprimento N` indica o comprimento da página em número de linhas (valor por omissão: 10 linhas);
- `-l M, --linha M` indica o número da primeira linha a mostrar (valor por omissão: linha 1).

3. Escreva um *script* que, à semelhança do comando `wc` (*word count*) do sistema `Unix`, conte o número de palavras, caracteres, e bytes em vários ficheiros de entrada. Adicionalmente, o *script* deverá aceitar as seguintes opções:

   - `-h, --help` mostra como utilizar o programa;
   - `-c` mostra o número de bytes do ficheiro;
   - `-l` mostra o número de linhas.

   Por exemplo:

```
$ python conta_palavras.py lab1.py lab2.py
  146   634   4404   lab1.py
  154   469   3781   lab2.py
  300   1103   8185   total

$ python conta_palavras.py -l lab1.py lab2.py
  146   lab1.py
  154   lab2.py
  300   total

$ python conta_palavras.py -c lab1.py lab2.py
  4404   lab1.py
  3781   lab2.py
  8185   total

$ python conta_palavras.py -l -c lab1.py lab2.py
  146   4404   lab1.py
```

```
  154   3781   lab2.py
  300   8185   total

$ python conta_palavras.py lab1.py lab2.py ff
  146   634   4404   lab1.py
  154   469   3781   lab2.py
conta_palavras.py: ff: open: No such file or directory
  300   1103   8185   total
```

4. Escreva um *script* para imprimir os ficheiros numa directoria e subdirectorias em forma de "árvore deitada". Para cada ficheiro, imprimos o seu nome; o nome de cada directoria deve ser precedido do sinal +. Cada nova directoria começa uma nova coluna dois espaços para a direita.

```
$ python pesquisa_arvore.py disciplinas/prog2
+docs
  exercicios.tex
  +trabalho
    enunciado.tex
    grafico.png
+src
  more.py
  deitados.py
```

5. Escreva um *script* que imprima o caminho absoluto de todos os ficheiros Python que se encontram numa dada directoria, incluindo as suas subdirectorias.

```
$ python todos_python.py exemplos
C:\temp\exemplos\scripts\more.py
C:\temp\exemplos\scripts\pesquisa_arvore.py
C:\temp\todos_python.py
```

   Escreva duas versões:

   (a) Versão iterativa, utilizando `os.walk()`;

   (b) Versão recursiva, utilizando `os.listdir()`.

   Sugestão: para obter o caminho absoluto utilize a função `os.path.abspath()`.

6. (a) Escreva um *script* que imprima o maior ficheiro numa dada diretoria, incluindo as suas subdiretorias.

```
$ python maior_ficheiro.py .
(18254877, 'atrasos.csv')
```

(b) Altere o *script* de modo a converter o tamanho desse ficheiro num número legível para humanos, seguindo as conversões:

- `1 GB = 1000 MB`;
- `1 MB = 1000 KB`;
- `1 KB = 1000 B`;

Exemplo:

```
$ python maior_ficheiro.py .
(18.25MB, 'atrasos.csv')
```

7. Este exercício tem como objectivo a escrita de *scripts* que imprimam o nome de todos os ficheiros Python numa certa diretoria que contenham alguma linha que emparelhe com uma certa expressão regular.

   (a) Por exemplo, o comando seguinte lista todos os ficheiros Python no diretoria `src` que tenham alguma linha iniciada pela palavra "import":

   ```
   $ python todos_regular.py '^import' src
   src/cat.py
   src/coop.py
   src/frequentes.py
   ```

   Sugestão: utilize a biblioteca para expressões regulares re, e adapte o exercício 5 acima.

   (b) Adapte o exercício anterior para que seja utilizada a diretoria corrente quando não for indicada a diretoria raiz.

   (c) Faça agora o seu *script* aceitar a opção `-r` indicando que a busca deve ser efectuada recursivamente nas subdiretorias.

8. Escreva um *script* que imprima a lista, sem repetidos e ordenada, de todos os módulos importados por todos os ficheiros Python constantes numa dada diretoria. Considere os padrões:

   - `import m1, ..., mn`
   - `from m import f`

   O nome de um módulo pode descrever submódulos utilizando um ponto: `from sound.effects.echo import echofilter`.

   ```
   >>> python todos_modulos.py sources
   [argparse, er, os, sys]
   ```

9. Escreva um *script* que dado um ficheiro comprimido no formato `.zip` mostre a lista de ficheiros nele contido, o tamanho original de cada ficheiro, e o seu tamanho comprimido.

```
$ python lista_zip.py umficheiro.zip

Nome do ficheiro         Tamanho              Compressão
t1-1415.pdf         300074 bytes    251683 bytes (83.9%)
t2-1415.pdf          83083 bytes     82260 bytes (99.0%)
```

Sugestão: utilize o módulo zipfile.