Folha de apoio à aula prática 9 (exercícios para praticar antes da aula).

9.1 Considere uma *lista recursiva* que contém inteiros ou listas recursivas. Defina uma função `ocorre(x,ys)` que testa se um valor `x` ocorre dentro de uma lista recursiva de inteiros `ys`. O resultado deve ser um valor de verdade. Exemplos:

```
>>> ocorre(1, [2,1,3])      >>> ocorre(0, [[2,1],3])      >>> ocorre(1, [[2,1],3])
True                        False                       True
```

9.2 Escreva uma função `dist(xs)` que retorna uma lista contendo todos os valores inteiros distintos presentes na lista recursiva de inteiros `xs`.

9.3 Escreva um programa que crie um ficheiro com o nome `"foo.txt"` com o seguinte conteúdo:

```
Hello from Python, text files!
```

9.4 Escreva um programa que leia o ficheiro com o nome `"foo.txt"` criado na pergunta anterior, e imprima o número de vezes que encontrou cada carater.

9.5 Escreva um programa que crie um ficheiro com o nome `"foo.pickle"` para guardar os seguintes valores:

```
a = ["valor de pi", math.pi]
b = ["valor de e", math.e]
c = {"pi": math.pi, "e": math.e}
```

9.6 Escreva um programa que leia o ficheiro com o nome `"foo.pickle"` criado na pergunta anterior, e imprima os valores aí guardados (restitua-os, por ordem, a variáveis com os nomes `a`, `b` e `c`).

9.7 Repita os dois exercícios anteriores, mas usando o formato "JSON". Com a ajuda de um editor, verifique que neste caso o ficheiro criado é um ficheiro de texto.

9.8 Implemente a função `words(filename)` que toma como argumento o nome de um ficheiro de texto e devolve um dicionário tal que: as chaves são as palavras existentes no ficheiro, e os valores são o número de vezes que aparecem. Pode assumir que o texto contém apenas palavras com carateres ASCII minúsculos, separadas por espaços.

9.9 A *sequência de Collatz* para um determinado valor $n > 0$ é obtida da seguinte forma: enquanto n não for igual a 1, gerar o próximo elemento da sequência usando a função:

$$f(n) = \begin{cases} n/2 & \text{se } n \text{ é par} \\ 3n+1 & \text{se } n \text{ é ímpar} \end{cases}$$

(a) Escreva uma função `it_collatz(n)` que retorna o número de iterações executadas para o valor `n`.

(b) Utilize-a para contruir um dicionário que, a cada inteiro n testado, associa o número de iterações utilizado.

(c) Implemente um programa que guarda esse dicionário num ficheiro. (Sugestão: utilize os módulos JSON ou Pickle.)

(d) Implemente um programa que lê esse dicionário, pergunta ao utilizador quantas iterações quer fazer (sejam k), efetua o cálculo para os k valores de n seguintes, e guarda o novo dicionário.

(e) Utilize este programa para, em várias sessões, obter o número de iterações para n de 1 a 1000000.

(f) Com base nesse dicionário, construa o histograma com a frequência de cada número de iterações observado nesse milhão de valores.

9.10 O código Morse associa letras do alfabeto ou algarismos a uma sequência de "pontos" e "traços". Considere a tabela seguinte:

```
1  .----  2  ..---  3  ...--  4  ....-  5  .....  6  -....
7  --...  8  ---..  9  ----.  0  -----
```

Escreva uma função `morse_number(filename)` que dado o nome `filename` de um ficheiro com pontos e traços correspondentes a algarismos, devolve o número inteiro correspondente.

Por exemplo, se o ficheiro `"cem.txt"` contiver o texto

```
.---- ----- -----
```

o resultado de `morse_number("cem.txt")` deverá ser 100.

Sugestão: começe por definir a tabela de código Morse como um dicionário.

9.11 O número de aluno numa conhecida universidade é constituído por nove algarismos, sendo os primeiros quatro relativos ao ano de inscrição, os dois seguintes o código do curso em que está inscrito, e o três últimos um código de identificação do aluno. Implemente a função `students(filename, year)` que dado o nome de um ficheiro `filename` com os inscritos na universidade, lê esse ficheiro o retorna um dicionário com o número de inscritos em cada curso no ano `year`.

Por exemplo, se o ficheiro `"inscritos.txt"` tiver o conteúdo

```
201701001
201801001
201801012
201801045
201802001
```

o resultado de `students("inscritos.txt", 2018)` é `{"01":3, "02":1}`

9.12 Em estatística, *moda* de um conjunto de valores é o valor que aparece mais vezes. Implemente a função `fmode(filename)` que dado o nome `filename` de um ficheiro de texto apenas com números inteiros, lê esse ficheiro o retorna uma lista com o(s) valor(es) da moda.

Por exemplo, se o ficheiro `"notas.txt"` tiver o conteúdo

```
10 9 15 10 15
```

o resultado de `fmode("notas.txt")` é `[10, 15]`.

9.13 Escreva uma função `spell_digits(filename)` que dado o nome `filename` de um ficheiro com algarismos, devolve uma *string* com a respetiva leitura em inglês separada por espaços. Os nomes dos dígitos em inglês são `zero one two three four five six seven eight nine`.

Por exemplo, se o ficheiro `"cem.dat"` contiver os dígitos

```
100
```

o resultado de `spell_digits("cem.dat")` deverá ser `"one zero zero"`.

9.14 Verificar se lista está contida noutra.

Defina a função `is_contained(a,b)` que verifica se todos os elementos da lista `a` estão contidos na lista `b`, devolvendo `True` nesse caso e `False` no caso contrário.

Por exemplo:

```
>>> is_contained([3,4,1],[1,2,4,1,3])
True
```

Na aula terá exercícios de avaliação contínua baseados nesta folha.