

Capítulo 8

Dicionários

1. Considere a seguinte lista de dicionários na qual os significados dos campos são óbvios:

```
l_nomes = [{'nome':{'nomep':'Jose', 'apelido':'Silva'},
'morada':{'rua':'R. dos douradores', 'num': 34, 'andar':'6 Esq',
'localidade':'Lisboa', 'estado':'', 'cp':'1100-032',
'pais':'Portugal'}}], {'nome':{'nomep':'John', 'apelido':'Doe'},
'morada':{'rua':'West Hazeltine Ave.', 'num': 57, 'andar':'',
'localidade':'Kenmore', 'estado':'NY', 'cp':'14317', 'pais':'USA'}}]
```

Diga quais são os valores dos seguintes nomes:

- (a) `l_nomes[1]`
 - (b) `l_nomes[1]['nome']`
 - (c) `l_nomes[1]['nome']['apelido']`
 - (d) `l_nomes[1]['nome']['apelido'][0]`
2. Escreva a função `agrupa_por_chave` que dada uma lista de pares chave valor (k, v) (representados por tuplos de dois elementos), devolve um dicionário que a cada chave k associa uma lista com os valores v para essa chave encontrados na lista passada como argumento. Exemplo:


```
>>> agrupa_por_chave([('a', 8), ('b', 9), ('a', 3)])
{'a': [8, 3], 'b': [9]}
```
 3. Uma carta de jogar é caracterizada por um naipe (espadas, copas, ouros e paus) e por um valor (A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K). Uma carta pode ser representada por um dicionário com duas chaves, `'np'` e `'vlr'`, sendo um conjunto de cartas representado por uma lista de cartas.
 - (a) Escreva uma função em Python que devolve uma lista contendo todas as cartas de um baralho. Por exemplo,

```
>>> baralho()
[{'np': 'esp', 'vlr': 'A'}, {'np': 'esp', 'vlr': '2'},
{'np': 'esp', 'vlr': '3'}, {'np': 'esp', 'vlr': '4'},
{'np': 'esp', 'vlr': '5'}, {'np': 'esp', 'vlr': '6'},
{'np': 'esp', 'vlr': '7'}, {'np': 'esp', 'vlr': '8'},
{'np': 'esp', 'vlr': '9'}, {'np': 'esp', 'vlr': '10'},
{'np': 'esp', 'vlr': 'J'}, {'np': 'esp', 'vlr': 'Q'},
{'np': 'esp', 'vlr': 'K'}, {'np': 'copas', 'vlr': 'A'},
{'np': 'copas', 'vlr': '2'}, {'np': 'copas', 'vlr': '3'},
... ]
```

- (b) Recorrendo à função `random()`, a qual produz um número aleatório no intervalo $[0, 1[$, escreva a função `baralha`, que recebe uma lista correspondente a um baralho de cartas e baralha aleatoriamente essas cartas, devolvendo a lista que corresponde às cartas baralhadas. SUGESTÃO: percorra sucessivamente as cartas do baralho trocando cada uma delas por uma outra carta seleccionada aleatoriamente. Por exemplo,

```
>>> baralha(baralho())
[{'np': 'esp', 'vlr': '3'}, {'np': 'esp', 'vlr': '9'},
{'np': 'copas', 'vlr': '6'}, {'np': 'esp', 'vlr': 'Q'},
{'np': 'esp', 'vlr': '7'}, {'np': 'copas', 'vlr': '8'},
{'np': 'copas', 'vlr': 'J'}, {'np': 'esp', 'vlr': 'K'},
... ]
```

- (c) Escreva uma função em Python que recebe um baralho de cartas e as distribui por quatro jogadores, devolvendo uma lista que contém as cartas de cada jogador. O seu programa deve garantir que o número de cartas a distribuir é um múltiplo de 4. Por exemplo,

```
distribui(baralha(baralho()))
[[{'np': 'ouros', 'vlr': 'A'}, {'np': 'copas', 'vlr': '7'},
{'np': 'paus', 'vlr': 'A'}, {'np': 'esp', 'vlr': 'J'},
{'np': 'paus', 'vlr': '6'}, {'np': 'esp', 'vlr': '10'},
{'np': 'copas', 'vlr': '5'}, {'np': 'esp', 'vlr': '6'},
{'np': 'copas', 'vlr': '8'}, {'np': 'esp', 'vlr': '3'},
{'np': 'ouros', 'vlr': '5'}, {'np': 'ouros', 'vlr': '8'},
{'np': 'copas', 'vlr': 'K'}],
[{'np': 'paus', 'vlr': '2'}, {'np': 'esp', 'vlr': '2'},
...]]
```

4. Escreva uma função, `metabolismo`, que recebe um dicionário cujas chaves correspondem a nomes de pessoas e cujos valores correspondem a tuplos, contendo o género, a idade, a altura e o peso dessa pessoa. A sua função devolve um dicionário que associa a cada pessoa o seu índice de metabolismo basal. Não é necessário validar os dados de entrada. Sendo s o género, i a idade, h a altura e p o peso de uma pessoa, o metabolismo

basal, m , é definido do seguinte modo:

$$m(s, i, h, p) = \begin{cases} 66 + 6.3 \times p + 12.9 \times h + 6.8 \times i & \text{se } s = M \\ 655 + 4.3 \times p + 4.7 \times h + 4.7 \times i & \text{se } s = F \end{cases}$$

Por exemplo:

```
>>> d = {'Maria': ('F', 34, 1.65, 64), 'Pedro': ('M', 34, 1.65, 64),
        'Ana': ('F', 54, 1.65, 120), 'Hugo': ('M', 12, 1.82, 75)}
>>> metabolismo(d)
{'Ana': 1458.955, 'Hugo': 675.078, 'Maria': 1109.755, 'Pedro': 736.685}
```

5. Escreva uma função que recebe uma cadeia de caracteres correspondente a um texto e que produz uma lista de todas as palavras que este contém, juntamente com o número de vezes que essa palavra aparece no texto. **SUGESTÃO:** guarde cada palavra como uma entrada num dicionário contendo o número de vezes que esta apareceu no texto. por exemplo,

```
>>> cc = 'a aranha arranha a ra a ra arranha a aranha ' \
        + 'nem a aranha arranha a ra nem a ra arranha a aranha'
>>> conta_palavras(cc)
{'aranha': 4, 'arranha': 4, 'ra': 4, 'a': 8, 'nem': 2}
```

6. Usando a ordenação por borbulhamento, escreva a função `mostra_ordenado` que apresenta por ordem alfabética os resultados produzidos pelo exercício anterior. Por exemplo,

```
>>> mostra_ordenado(conta_palavras(cc))
a 8
aranha 4
arranha 4
nem 2
ra 4
```

7. Uma matriz é dita *esparsa* (ou rarefeita) quando a maior parte dos seus elementos é zero. As matrizes esparsas aparecem em grande número de aplicações em engenharia. Uma matriz esparsa pode ser representada por um dicionário cujas chaves correspondem a tuplos que indicam a posição de um elemento na matriz (linha e coluna) e cujo valor é o elemento nessa posição da matriz. Por exemplo, $\{(3, 2): 20, (150, 2): 6, (300, 10): 20\}$ corresponde a uma matriz esparsa com apenas três elementos diferentes de zero.

- (a) Escreva uma função em Python que recebe uma matriz esparsa e a escreve sob a forma, na qual os elementos cujo valor é zero são explicitados.

$$\begin{array}{cccc} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \cdots & \cdots & & \cdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{array}$$

```
>>> escreve_esparsa({(1,5): 4, (2, 3): 9, (4, 1): 1})
0 0 0 0 0 0
0 0 0 0 0 4
0 0 0 9 0 0
0 0 0 0 0 0
0 1 0 0 0 0
```

- (b) Escreva uma função em Python que recebe duas matrizes esparsas e devolve a sua soma. Por exemplo,

```
e1 = {(1,5): 4, (2, 3): 9, (4, 1): 1}
e2 = {(1, 6): 2, (4, 1): 2, (5,4): 2}
>>> escreve_esparsa(soma_esparsa(e1, e2))
0 0 0 0 0 0 0
0 0 0 0 0 4 2
0 0 0 9 0 0 0
0 0 0 0 0 0 0
0 3 0 0 0 0 0
0 0 0 0 2 0 0
```

8. Suponha que `bib` é uma lista cujos elementos são dicionários e que contém a informação sobre os livros existentes numa biblioteca. Cada livro é caracterizado pelo seus autores, título, casa editora, cidade de publicação, ano de publicação, número de páginas e ISBN. Por exemplo, a seguinte lista corresponde a uma biblioteca com dois livros:

```
[{'autores': ['G. Arroz', 'J. Monteiro', 'A. Oliveira'],
'titulo': 'Arquitectura de computadores', 'editor': 'IST Press',
'cidade': 'Lisboa', 'ano': 2007, 'numpags': 799,
'isbn': '978-972-8469-54-2'}, {'autores': ['J.P. Martins'],
'titulo': 'Logica e Raciocinio', 'editor': 'College Publications',
'cidade': 'Londres', 'ano': 2014, 'numpags': 438,
'isbn': '978-1-84890-125-4'}]
```

Escreva um programa em Python que recebe a informação de uma biblioteca e devolve o título do livro mais antigo. Por exemplo:

```
>>> mais_antigo(bib)
'Arquitectura de computadores'
```

9. Um tabuleiro de xadrez tem 64 posições organizadas em 8 linhas e 8 colunas. As linhas são numeradas de 1 a 8 e as colunas de A a H, sendo a posição inferior esquerda a 1, A. Neste tabuleiro são colocadas peças de duas cores (brancas e pretas) e de diferentes tipos (rei, rainha, bispo, torre, cavalo e peão). Um tabuleiro de um jogo de xadrez pode ser representado por um dicionário cujos elementos são do tipo $(l, c): (cor, t)$. Por exemplo o elemento do dicionário $(2, C): (branca, rainha)$ indica que a rainha branca está na segunda linha, terceira coluna.

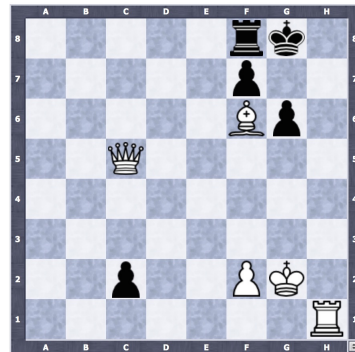


Figura 8.1: Tabuleiro de xadrez.

A situação do jogo de xadrez apresentado na Figura 8.1 é representada pelo seguinte dicionário:

```
j = {(1, 'H'): ('branca', 'torre'), (2, 'F'): ('branca', 'peao'), \
      (2, 'G'): ('branca', 'rei'), (6, 'F'): ('branca', 'bispo'), \
      (5, 'C'): ('branca', 'rainha'), (6, 'G'): ('preta', 'peao'), \
      (7, 'F'): ('preta', 'peao'), (8, 'F'): ('preta', 'torre'), \
      (8, 'G'): ('preta', 'rei'), (2, 'C'): ('preta', 'peao')}
```

Num jogo de xadrez, a rainha movimenta-se na vertical, na horizontal ou nas diagonais e pode atacar qualquer peça da cor contrária que possa ser atingida num dos seus movimentos, desde que não existam outras peças no caminho. Escreva uma função em Python que recebe um tabuleiro de xadrez e determina quais as peças que podem ser atacadas pelas rainhas. Por exemplo,

```
>>> ataques_rainhas(j)
[['peao', (2, 'C')], ['torre', (8, 'F')]]
```

10. Tendo em conta a representação de um tabuleiro de xadrez apresentada no exercício anterior, escreva um programa que recebe a posição de um cavalo e determina os seus movimentos possíveis. Um cavalo move-se andando duas posições para a frente e uma para o lado. Note-se que um cavalo pode mover-se para uma casa onde exista uma peça do adversário mas não se pode mover para uma casa em que exista uma peça com a sua cor.