# Ficha de trabalho #1

***Dicionários***

1. Um snack-bar tem o seguinte preçário:

|  |  |
| --- | --- |
| **Produto** | **Preço** |
| Salgados | 0,60 € |
| Lanche | 1,20 € |
| Sumo | 1,50 € |
| Refrigerante | 1, 00 € |
| Bolo | 0,70 € |

* 1. Crie um dicionário para armazenar o preçário do snack-bar.
  2. Utilizando o dicionário criado na alínea anterior, calcule qual seria o preço a pagar se o cliente pedisse um lanche, um bolo e um sumo de laranja.
  3. Desenvolva um programa que:
* questione o pedido do cliente (até o cliente introduzir “stop”)
* faça o cálculo do valor total do pedido
* pergunte com quanto o cliente irá pagar
* calcule o troco a devolver.

Exemplo:

|  |  |
| --- | --- |
| **Input:** | **Output:** |
|  | Introduzir produtos. Para parar: stop |
| lanche  bolo  stop | Total: 1.9€  com quanto paga? |
| 5 | Troco = 3.1€ |

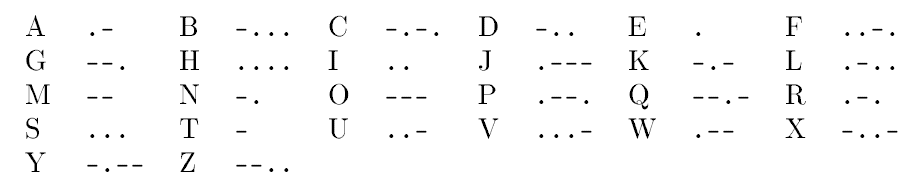
* 1. Crie uma função que use o dicionário para calcular a média dos preços dos produtos.

1. Escreva uma função traduz que recebe como argumentos uma lista de palavras e um dicionário e devolve uma nova lista de palavras traduzidas usando a tradução do dicionário. Caso a lista de palavras contenha uma palavra que não exista no dicionário, essa palavra deverá ficar por traduzir. O output deve ser mostrado como uma frase completa[[1]](#footnote-1).

Exemplo:

|  |  |
| --- | --- |
| **Chamada função:** | **Output:** |
| pt\_en={"hoje":"today",  "esta":"is",  "enevoado":"cloudy"}  txt= ["hoje","esta","muito","enevoado"]  traduz(txt, pt\_en) | today is muito cloudy |

1. O código Morse associa cada letra do alfabeto a uma sequência de “pontos” e “traços”



* 1. Defina a tabela de código Morse como o seguinte dicionário:

code = {"A": ".-", "B": "-...", "C": "-.-", "D": "-..", "E": ".", "F": "..-.", "G": "--.", "H": "....", "I": "..", "J": ".----","K": "-.-","L": ".-..", "M": "--", "N": "-.", "O": "----", "P": ".--.", "Q": "--.-", "R": ".-.", "S": "...", "T": "-", "U": "..-","V": "-.---", "Z": "--.."}

* 1. Implemente a função morse(txt) que converte as letras numa sequência de carateres para Morse; o resultado deve ser uma cadeia com pontos e traços. Usar um espaço para separar sequências correspondentes às letras. Os carateres do texto original que não forem letras maiúsculas devem ser ignorados.

Exemplos:

|  |  |
| --- | --- |
| **Chamada função:** | **Output:** |
| morse("ABC") | .- -... -.- |
| morse("A B C") | .- -... -.- |
| morse("ABC xyz") | .- -... -.- |
| morse("ATTACK AT DAWN") | .- - - .- -.- -.- .- - -.. .- -. |

1. No campeonato nacional de futebol uma vitória conta 3 pontos, um empate 1 ponto e uma derrota 0 pontos. Em cada jogo, ganha a equipa que marcar mais golos, havendo um empate se o número de golos for o mesmo.
   1. Implemente a função futebol(scores) que retorna um dicionário com a pontuação de cada equipa no final da temporada. O parâmetro scores é uma lista de dicionários com os resultados da jornada. Cada dicionário tem nome de um clube na chave e o número de golos no valor. Por exemplo,

Exemplo:

|  |  |
| --- | --- |
| **Chamada função:** | **Output:** |
| futebol ([  {"Vitória SC":2, "Boavista":1},  {"Gil Vicente":1, "Rio Ave":1},  {"Famalicão":3, "Sporting":2},  {"FC Porto":0, "Benfica":0},  {"Tondela":2, "Santa Clara":3}  ]) | {  'Vitória SC': 3,  'Boavista': 0,  'Gil Vicente': 1,  'Rio Ave': 1,  'Famalicão': 3,  'Sporting': 0,  'FC Porto': 1,  'Benfica': 1,  'Santa Clara': 3,  'Tondela': 0  } |

* 1. Acrescente o código seguinte ao desenvolvido e verifique o que acontece

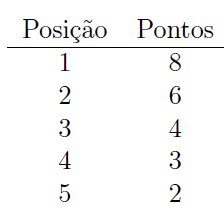
res = futebol(scores)

import operator

res = dict( sorted(res.items(), key=operator.itemgetter(1), reverse=True))

print(res)

1. Nos anos 50 do século passado, no campeonato de Fórmula 1 a pontuação de cada prova era atribuída com base na posição de cada corredor no final, de acordo com a seguinte tabela:



* 1. Implemente a função formula1(scores) que, dada uma lista com a lista ordenada dos cinco melhor classificados, para cada prova de uma época, retorna um dicionário com a pontuação de cada corredor. Por exemplo,

Exemplo:

|  |  |
| --- | --- |
| **Chamada função:** | **Output:** |
| formula1([  ['Sainz', 'Verstappen', 'Hamilton', 'Ricciardo', 'Massa'],  ['Bottas', 'Verstappen', 'Raikkoten', 'Stroll', 'Vettel'],  ['Perez', 'Raikkoten', 'Verstappen', 'Hamilton', 'Vettel']  ]) | {  'Sainz': 8,  'Verstappen': 16,  'Hamilton': 7,  'Ricciardo': 3,  'Massa': 2,  'Bottas': 8,  'Raikkoten': 10,  'Stroll': 3,  'Vettel': 4,  'Perez': 8  } |

* 1. Acrescente o código do exercício 4.2 e verifique o que é alterado.

1. Este exercício tem como objetivo simular o inventário de uma loja de informática.
   1. Utilizando dicionários e listas, construa o inventário da loja. O inventário tem de conter no mínimo 10 produtos. A informação a incluir para cada produto é: nome do produto, quantidade e preço.
   2. Implemente a função compra(listaCompras) que

* recebe um dicionário com a lista das compras (do tipo: {produto1: quantidade1, …})
* caso os produtos existam no inventário e haja quantidade suficiente para a venda, determinam o preço final dos produtos
* caso não existam, ou o stock seja insuficiente, mostre uma mensagem de acordo
* determine o preço total da compra

Exemplo de output:

Stock insuficiente do produto \*rato sem fios\*

preço (3x teclado [7.5€/un]):22.5€

Produto \*batatas fritas\* não existe

Total compra: 22.5€

1. <https://www.w3schools.com/python/ref_string_join.asp> [↑](#footnote-ref-1)