GUIA 2 - EJERCICIOS UNIDAD 4

FUNCIONES – LISTAS – DICCIONARIOS

1.

La banda de rock Strip'n Split ha concluído su gira mundial. La información de cada uno de sus conciertos está almacenada en una lista en orden cronológico.

Cada elemento de la lista es un diccionario con tres llaves: `'ciudad'`, `'publico'` y `'canciones'`. El valor asociado a `'publico'` es la cantidad de personas que asistió al concierto, y el asociado a `'canciones'` es la lista de las canciones que fueron tocadas en ese concierto.

a) Escriba una función llamada `cuantos_escucharon(c)`, que retorne la cantidad de personas que escucharon la canción `c`.

b) Escriba una función llamada `mismo_concierto(c1, c2)`, que retorne **`True`** o **`False`** para indicar si las canciones `c1` y `c2` fueron tocadas alguna vez en el mismo concierto.

```
conciertos = [
  {
    'ciudad': 'Cairo',
    'publico': 30000,
    'canciones': [
      'If your name is main',
      'Break my heart and return',
      'Strings float around you',
    ]
  },
  {
    'ciudad': 'Moscu',
    'publico': 25000,
    'canciones': [
      'Break my heart and return',
      'Open your mind, close your file',
      'Value error'
    ]
  }
  # ...
]
```

2.

(Este ejercicio es una continuación del anterior).

El diccionario `ciudades` asocia a cada país una lista de las ciudades de ese país en las que tocó Strip'n Split:

```
ciudades = {
  'Suiza': ['Zurich', 'Ginebra'],
  'Israel': ['Tel Aviv'],
  'Marruecos': ['Rabat', 'Casablanca'],
  'China': ['Shanghai', 'Beijing'],
  # ...
}
```

a) Escriba la función `canciones_pais(p)`, que retorne el conjunto de las canciones que fueron tocadas en el país `p`.

b) Escriba la función `contar_exitos(n)`, que retorne la cantidad de veces en que hubo dos conciertos consecutivos en el mismo país que tuvieron más de `n` espectadores.

3.

El mundo está sumido en el caos. Una epidemia ha arrasado con gran parte de la población. Los humanos infectados regresan como *walkers*, muertos vivientes hambrientos de carne humana. Sin embargo, los walkers no son el único peligro: muchas veces, los sobrevivientes deben enfrentarse entre ellos para mantenerse con vida.

En este mundo post-apocalíptico, tu misión es desarrollar una aplicación en Python para **clasificar a los miembros de tu grupo de sobrevivientes**.

La información del grupo se encuentra en el siguiente **diccionario**, donde la **clave** es el nombre del miembro del grupo y el **valor** es una **lista** con dos números: la cantidad de *walkers* y la cantidad de humanos eliminados por ese miembro.
Por ejemplo, Rick ha matado 172 walkers y 10 humanos:

```
grupo = {
   'rick': [172, 10], 'daryl': [136, 11], 'michonne': [80, 6],
   'glenn': [73, 0], 'maggie': [55, 4], 'carl': [62, 1],
   'tyreese': [35, 0], 'carol': [17, 3]
}
```

a) Función `total(grupo)`

Desarrolla una función que reciba como parámetro un diccionario con las estadísticas del grupo y retorne una **tupla** con el total de walkers y humanos eliminados por todo el grupo.

```
>>> total(grupo)
(630.0, 35.0)
```

b) Función `puntaje(grupo)`

Desarrolla una función que reciba el diccionario del grupo y retorne un nuevo diccionario, donde la clave sea el nombre del miembro y el valor su puntaje.
El puntaje se calcula como:

```
walkers / total_walkers + 2 * (humanos / total_humanos)
```

Si un valor total es cero, se debe evitar la división por cero.

```
>>> puntaje(grupo)
{'maggie': 0.32, 'glenn': 0.12, 'michonne': 0.47, 'rick': 0.84,
 'carl': 0.16, 'carol': 0.2, 'daryl': 0.84, 'tyreese': 0.06}
```

c) Función `ranking(grupo)`

Desarrolla una función que reciba el diccionario del grupo y retorne una **lista ordenada de mayor a menor puntaje** con los nombres de los miembros.

```
>>> ranking(grupo)

['rick','daryl','michonne','maggie','carol','carl','glenn','tyreese']
```