Laboratorio UVA-8

1) Para analizar datos de **Series de Televisión** se cuenta con la información de todas las series en una lista de listas en donde cada lista tiene la siguiente estructura: [nombre, pais de origen, rating, géneros]. A su vez, géneros es una lista con todos los géneros a los cuales pertenece dicha serie. Por ejemplo:

series = [
['Game of thrones','USA',9.4,['ficcion']],
['24','USA',8.4,['accion','suspenso']],
['La casa de papel','Espa˜na',9.2,['accion','suspenso','drama']],
['Orange is the new black', 'USA', 8.5, ['comedia','drama']],
['Dark', 'Alemania', 9.2, ['ficcion','drama']],
['Sherlock','UK',9.2,['policial','drama','suspenso']],
['Merl´ı','Espa˜na',9.5,['drama']],
['Whitecollar','USA',8.2,['comedia','drama','suspenso']],
['Heroes','USA',7.7,['ficcion','accion']],
['Mistfit','UK',8.4,['accion','drama','ficcion']]
]

Observe que la serie 24 es de USA, tiene un rating de 8.4 y está catalogada como de acción y suspenso.

(a) Escriba la función **series_x_genero**(series), que recibe como parámetro la lista de series y retorna un diccionario en el que las llaves son los distintos géneros y los valores son los nombres de las series asociadas a cada género, ordenados alfabéticamente.

print(series_x_genero(series))

{'ficcion': ['Dark', 'Game of thrones', 'Heroes', 'Mistfit'], 'accion': ['24', 'Heroes', 'La casa de papel', 'Mistfit'], 'suspenso': ['24', 'La casa de papel', 'Sherlock', 'Whitecollar'], 'drama': ['Dark', 'La casa de papel', 'Merl´ı', 'Mistfit', 'Orange is the new black', 'Sherlock', 'Whitecollar'], 'comedia': ['Orange is the new black', 'Whitecollar'], 'policial': ['Sherlock']}

(b) Escriba la función **países_con_mas_series(series)**, que recibe como parámetro la lista de series y retorna una lista de strings con los nombres de los 3 países que tienen más series producidas, ordenada de mayor a menor de acuerdo al número de series. Puede suponer que siempre habrá suficientes países, y que no habrá empates entre los países en la cantidad de series. Utilice un diccionario para completar la tarea de contar.

print(paises_con_mas_series(series))

['USA', 'UK', 'España']

2) La conocida aplicación **SpotiPhy** necesita determinar cuáles son los álbumes y los artistas mejores evaluados en su plataforma. Para lograrlo dispone de una lista de listas que tienen la siguiente forma: [canción, album, artistas, fecha evaluación, nota], en donde canción y álbum están en formato string, artistas es una lista que contiene los nombres de los artistas (una canción puede tener más de un artista) en formatro string, la fecha es una lista [A,M,D] y la nota es un entero entre 1 y 5. A continuación se muestra un ejemplo de la estructura evaluaciones:

evaluaciones = [
  ['Phypocrita', 'Hasta abajo', ['Pynuel'], [2018, 6, 30], 4],
  ['Sin Phyjama', 'Nuevo Estilo', ['Becky T', 'Turize'], [2018, 4, 14], 3],
  ['Vaina Crazy', 'del Weno', ['Ozune', 'Turize'], [2018, 7, 18], 5],
  ['Phypocrita', 'Hasta abajo', ['Pynuel'], [2018, 8, 20], 5],
  ['Quiero Beber Agua', 'Hasta abajo', ['Pynuel'], [2018, 2, 25], 5]
]

Notar que una misma canción pudo ser evaluada en distintos momentos. De acuerdo con la información anterior, se le solicita implementar las siguientes funciones:

(a) Escriba la función **notas_por_artista(lista)**, donde lista es del tipo evaluaciones. Esta función debe retornar un diccionario donde la llave es un string con el nombre del artista y como valor tienen una lista de enteros que corresponden a todas las evaluaciones de canciones en las que ha participado. Recuerde que una misma canción puede tener varias evaluaciones y todos estos valores deben estar en la lista.

print notas_por_artista(evaluaciones)

{'Ozune': [5], 'Pynuel': [4, 5, 5], 'Becky T': [3], 'Turize': [3, 5]}

b) Escriba la función artistas_hit(lista, fecha), donde lista es del tipo evaluaciones y fecha es una lista con formato [A,M,D]. La función debe retornar una lista con los 3 artistas que con mayor probabilidad de producir hits, ordenada de forma descendente según la probabilidad. Esta probabilidad se obtiene al dividir la cantidad de evaluaciones con nota 4 o 5 del artista por la cantidad total de evaluaciones que posee dicho artista de sus canciones, en ambos considerar las evaluaciones hasta la fecha consultada, inclusive. Por lo tanto, la probabilidad siempre será un número entre 0 y 1. Si hay menos de 3 artistas con evaluaciones anteriores a la fecha, entregue una lista con todos los artistas posibles. Tenga en consideración que los 3 mejores artistas seleccionados deben tener probabilidad mayor a todo el resto de los artistas en SpotiPhy.

print artistas_hit(evaluaciones,[2018,8,30])

['Pynuel', 'Ozune', 'Turize’]

print artistas_hit(evaluaciones,[2018,2,28])

['Pynuel']

3) PyCornerShop es una aplicación que le permite a sus usuarios conseguir productos del supermercado desde la comodidad del hogar usando sus teléfonos móviles. La información de los repartidores y usuarios se encuentra almacenada en dos diccionarios llamados repartidores y usuarios respectivamente. **El diccionario repartidores** tiene los nombres de los repartidores como llave y como valor una lista que contiene la ubicación del repartidor en el plano de la ciudad y su estado de disponibilidad: True (disponible) o False (no disponible). A continuación, se muestra un ejemplo de este diccionario**:**

repartidores = { 'rayo macuin': [[10, 2], True], 'reparti dhor': [[9, 3], True], 'eliseo al-azar': [[5, 5], False]}

El diccionario usuarios contiene el código del usuario como llave y como valor la ubicación del mismo en el plano de la ciudad. A continuación, se muestra un ejemplo:

usuarios = {1221: [5, 2], 441: [8, 2], 587: [10, 1]}

Existe un diccionario llamado visitas donde cada llave es el nombre de un repartidor y como valor tiene una lista de listas. Cada lista se compone de un código de usuario y de la cantidad de veces que dicho repartidor (llave del diccionario) visito a dicho usuario. Tenga en consideración que, si un usuario nunca ha solicitado reparto, no aparecerá en el diccionario visitas.

visitas = {
  'rayo macuin': [[1221, 5], [441, 8], [587, 2]],
  'reparti dhor': [[1221, 2], [441, 5], [587, 3]],
  'eliseo al-azar': [[1221, 8], [441, 2], [587, 1]]
}

Para que la compañia PyCornerShop pueda funcionar de manera eficiente, le solicita a Ud. que implemente una función llamada **buscar_repartidor(repartidores,usuarios,visitas,codigo)** que reciba como parámetros los diccionarios antes mencionados y un código de usuario (formato int). La función debe retornar una lista de repartidores disponibles y cercanos, ordenados de manera descendente, según el número de visitas que estos hayan realizado al usuario requerido. **Un repartidor será considerado cercano si está ubicado a menos de 4 km del usuario** y para esto considere que las listas de ubicación tienen valores enteros medidos en km. Finalmente, si no hay repartidores cercanos, la función debe retornar una lista vacía.

print buscar_repartidor(repartidores,usuarios,visitas,587)
['reparti dhor', 'rayo macuin']
print buscar_repartidor(repartidores,usuarios,visitas,441)
['rayo macuin', 'reparti dhor']
print buscar_repartidor(repartidores,usuarios,visitas,1221)
[]

Para calcular la distancia d entre el usuario ubicado en [x1,y1] y un repartidor ubicado en [x2,y2] se debe utilizar la ecuación: d = $\sqrt{(x1 - x2)^2 + (y1 - y2)^2}$