**MusTic**

La empresa de música en streaming MusTic ha incrementado significativamente la cantidad de canciones en su catálogo durante los últimos meses, por lo que ha tenido problemas para gestionar su información.

La empresa, en sus bases de datos internas representa cada canción con un número entero y con esta información es usual que se realicen tareas como, eliminar duplicados sobre un listado de música, encontrar canciones que pertenecen a ciertas categorías, permitir a sus usuarios intercambiar música, entre otras.

Hasta el momento, todas estas tareas se realizan de forma manual por lo que la empresa quiere empezar a usar los beneficios de la programación para que todas estas se puedan realizar de forma automática.  
  
Usted es contratado para realizar un módulo/librería de Python que tenga las funciones requeridas para ejecutar automáticamente las tareas de procesamiento de información realizadas por la empresa.

Se describen a continuación, cada una de las funciones que deben estar incluidas el módulo/librería que usted va a desarrollar.

Nombre del módulo/librería: utilsmusic

Funciones que debe contener la librería “utilsmusic”

1.      Nombre de la función: canciones\_sin\_repeticion  
Descripción:  la función debe recibir una lista de canciones E1 (representadas por números enteros) y debe eliminar las que se encuentren repetidas, retornando una lista donde cada canción aparece una sola vez.

Entradas:

·        E1: (lista) Lista de canciones

Retorno:

·        (lista) Lista de canciones sin repetición

Ejemplo:

El llamado a la función canciones\_sin\_repeticion ( [8, 7, 8, 2, 3, 2, 1, 3] )  
debería retornar [8, 7, 2, 3, 1]

2.      Nombre de la función: posiciones\_con\_cancion  
Descripción:  A partir de una canción E3, una lista de canciones E2 y una lista de posiciones E1, se desea conocer, de la lista de posiciones E1, cuáles de estas posiciones corresponden a la canción E3 si buscamos en la lista de música E2 en dicha posición.  La función debe retornar una lista con los elementos de E1 que al acceder a ellos (a esa posición) en E2 correspondan a la canción E3. Ver ejemplo.

Entradas:

·        E1: (lista) Lista de posiciones

·        E2: (lista) Lista de canciones

·        E3: (entero) Canción

Retorno:

·        (lista) Lista de posiciones seleccionadas de acuerdo con la descripción

Ejemplo:

El llamado a la función posiciones\_con\_cancion ( [1, 2, 5], [4, 4, 6, 2, 1, 4], 4 ) debería retornar [1, 5], porque la canción**4** se encuentra en la posición 1 y también en la posición 5 pero NO en la posición 2 de la lista de canciones.  
El llamado a la función posiciones\_con\_cancion( [0, 2, 3, 6], [7, 8, 9, 7, 9, 9, 9, 8], 9 ) debería retornar [2, 6], porque la canción **9** se encuentra en la posición 2 y también en la posición 6 pero NO en la posición 0 ni en la 3 de la lista de canciones.

3.      Nombre de la función: solo\_categoria\_X  
Descripción:  A partir de la lista de música de categoría X (E1) y la lista de música de categoría Y (E2) se desea conocer las canciones que pertenecen únicamente a la categoría X. La función debe retornar una lista con la música que pertenecen **únicamente** a la categoría X.

Entradas:

·        E1: (lista) Lista de música de categoría X

·        E2: (lista) Lista de música de categoría Y

Retorno:

·        (lista) Lista de música que pertenecen únicamente a la categoría X

Ejemplo:

El llamado a la función solo\_categoria\_X ( [5, 1, 4], [2, 1] )  
debería retornar [5, 4]

El llamado a la función solo\_categoria\_X ( [3,5,7,10,15,16], [4,10,5,8] )  
debería retornar [3,7,15,16]

4.      Nombre de la función: cantidad\_cambios  
Descripción: La empresa también permite a sus usuarios intercambiar canciones que ya no escuchan. Por lo tanto, se requiere que a partir de las listas de canciones que ya no escuchan dos usuarios (lista E1 y lista E2) se calcule el número de canciones que pueden intercambiar estos dos usuarios entre ellos. La función debe recibir las listas de canciones que ya no escuchan dos usuarios y debe retornan la cantidad de música que pueden intercambiar, teniendo en cuenta que el cambio se hace 1 a 1 (se cambia 1 canción por otra) y que los dos están interesados en cualquier canción menos en las que están en su propia lista.

Entradas:

·        E1: (lista) Lista de música que el usuario1 ya no escucha

·        E2: (lista) Lista de música que le usuario2 ya no escucha

Retorno:

·        (entero) cantidad de canciones que pueden intercambiar

Ejemplo:

El llamado a la función cantidad\_cambios( [3, 5, 7, 10, 15, 16], [4, 10, 5, 8])  
debería retornar 2, porque al primer usuario solo le interesan dos canciones del segundo usuario: [4,8] (las que él no tiene pero el otro sí), mientras que al segundo usuario solo le interesan las canciones: [3,7,15,16]. Por lo tanto, cómo al primer usuario le interesan 2 canciones y al segundo le interesan 4, únicamente se pueden realizar 2 intercambios.

Realice un módulo/librería en Python que contenga las 4 funciones descritas previamente.

IMPORTANTE: el nombre del módulo/librería y de las funciones debe ser igual al que está descrito en las especificaciones previas.

Calificación:

El juez comprobará la existencia del módulo/librería y ejecutará varios casos de prueba por cada una de las 4 funciones definidas. La calificación será sobre 5 puntos que se obtienen de la siguiente forma:

1 punto: Existencia del módulo/librería “utilsmusic”  (debe crear un archivo con el nombre utilsmusic.py que contenga las funciones)  
1 punto: Definición apropiada de la función “canciones\_sin\_repeticion” dentro del módulo  
1 punto: Definición apropiada de la función “posiciones\_con\_cancion” dentro del módulo  
1 punto: Definición apropiada de la función “solo\_categoria\_X” dentro del módulo  
1 punto: Definición apropiada de la función “cantidad\_cambios” dentro del módulo