Ejercicios clase: Funciones 2

1. Haz una función lambda que tome como entrada una Lista y sume todas las componentes. Podeis utilizar sum().
    a. Aplica esta función a una lista de listas con la función map.

2. Haz una función lambda que tome como entrada una string y devuelva True si el string empieza en mayúscula.
    a. Aplica esta función a una lista de nombres y descarta los que no empiezan con mayúscula con la función filter.

3. Haz una función que calcula la suma armónica hasta un número entero n. Utiliza la recursividad.

$$1+\frac{1}{2}+\frac{1}{3}+\frac{1}{4}+\frac{1}{5}+\cdots$$

4. Haz una función que dada una lista de listas (donde dentro puede haber más listas) te devuelva la suma de todos los elementos. Utiliza recursividad. [Ayuda: type([1,2,3]) == list #True,]

5. Dado un grafo no dirigido (lista de parejas de id_vertices conectados), y dos id_vertices, escribe una función que devuelva true si están conectados, es decir, si existe un camino entre ellos. Por ejemplo para el grafo = [[1,2], [2,3], [2,4], [5,6]] los id_vertices [1, 4] están conectados y los id_vertices [1,5] no lo están. Sigue los siguientes pasos:
    a. Haz una función que, dado un id_vertice cualquiera, devuelva los id_vertices vecinos a este.
    b. Ahora piensa la forma de iterar este proceso [Pista: usa un while hasta que llegues al id_vertice que buscas o hasta que no queden caminos por explorar]

6. Crea una función que, dado un sudoku en una lista (lista de 81 elementos), devuelva true si este sudoku es correcto. Los pasos para desarrollar la solución son los siguientes [Pista: Puede ser una buena idea como primer paso transformar la lista a una matriz]:
    a. Crea una función para devolver el vector fila en el que está el j-esimo elemento del sudoku j = 1, …, 81

b. Crea una función para devolver el vector columna en el que está el j-esimo elemento del sudoku j = 1, …, 81
c. Crea una función para devolver el subcuadrado 3x3 en el que está el j-esimo elemento del sudoku j = 1, …, 81.
d. Toma el primer elemento del sudoku y crea funciones para verificar:
    i. que no haya ningún número igual en la misma fila
    ii. que no haya ningún número igual en la misma columna
    iii. que no haya ningún número igual en el cuadrado 3 x 3 al que pertenece
e. Ahora solo tienes que recorrer el sudoku entero

Este es un ejemplo de un sudoku correcto:
sudoku = [4, 3, 5, 2, 6, 9, 7, 8, 1,
6, 8, 2, 5, 7, 1, 4, 9, 3,
1, 9, 7, 8, 3, 4, 5, 6, 2,
8, 2, 6, 1, 9, 5, 3, 4, 7,
3, 7, 4, 6, 8, 2, 9, 1, 5,
9, 5, 1, 7, 4, 3, 6, 2, 8,
5, 1, 9, 3, 2, 6, 8, 7, 4,
2, 4, 8, 9, 5, 7, 1, 3, 6,
7, 6, 3, 4, 1, 8, 2, 5, 9]