Clase 09. Funcional 1

Ejercicio 1:

Utilizando la función reduce, escribir una función que tenga como entrada una lista de palabras y retorne la palabra mas larga.
Ídem pero que retorne la menor alfabéticamente

```
lista_palabra = ['hola','como','andannnnn','todos','a']

import functools

def compar_palabra(palabra1,palabra2):
    if len(palabra1) > len(palabra2):
        return palabra1
    else:
        return palabra2

def compar_palabra_alfabetico(palabra1,palabra2):
    return palabra1 if palabra1 < palabra2 else palabra2

functools.reduce(compar_palabra_alfabetico,lista_palabra)
```

Ejercicio 2:
Utilizando la función map, escribir una función que dada una lista de palabras retorne una lista de tuplas donde cada tupla tenga este formato: (palabra, largo de la palabra)

```
lista_palabra = ['hola','como','andannnnn','todos','a']

def tupla_palabra(palabra):
    return (palabra,len(palabra))

list(map(tupla_palabra,lista_palabr
```

Ejercicio 3:
Utilizando la función filter, escribir una función que tenga como entrada una lista de palabras y un entero n y que retorne una lista con las palabras de largo mayor a n.

```
lista_palabra = ['hola','como','andannnnn','todos','a']

def es_mayor_3_aux(palabra):
    return len(palabra) > 3

def ejercicio3(max,lista_palabra):
    print(list(filter(es_mayor_3_aux,lista_palabra)))
```

Ejercicio 4

Usar filter para que, dado un string, se devuelva una lista con todas las vocales contenidas en él

```
def esVocal(letra):
    return letra.lower() in "aeiou"

print(list(filter(esVocal,"asdjglkjgp goieeoi")))
```

Ejercicio 5

En una biblioteca tenemos la siguiente información acerca de órdenes de compra.

Id	Título y autor	Cantidad	Precio/unidad
34587	Learning Python, Mark Lutz	4	40.95
98762	Programming Python, Mark Lutz	5	56.80
77226	Head First Python, Paul Barry	3	32.95

Escribir un programa que retorne una lista con tuplas de 2 elementos.

Cada tupla consiste en el id de la orden (primer elemento) y el precio (de acuerdo a la cantidad).

El precio final debe tener un descuento de 10% en caso de que la orden supere los 100. La información relativa a las ordenes está representada como una lista de listas:

orders = [["34587", "Learning Python, Mark Lutz", 4, 40.95],
["98762", "Programming Python, Mark Lutz", 5, 56.80],
["77226", "Head First Python, Paul Barry", 3,32.95]]

```
def tupla(lista):
    valor = lista[2]*lista[3]
    if valor > 100:
        valor = valor*0.9
    return(lista[0], round(valor,2))

orders = [ ["34587", "Learning Python, Mark Lutz", 4, 40.95], ["98762", "Programming Python, Mark Lutz", 5,
56.80], ["77226", "Head First Python, Paul Barry", 3,32.95]]

print(list(map(tupla, orders)))
```

Opción con map de map:

```
def f1(l):
    return (l[0], l[2]*l[3])

def f2(x):
    if x[1] >= 100:
        return (x[0], round(x[1]*0.9,2))
    else:
        return (x[0], round(x[1],2))

invoice_totals_2 = list(map(f2, map(f1, orders)))
```