# EJERCICIOS PYTHON - MISCELÁNEA

## 1. IT_IMPARES.PY

Diseña un generador que devuelva una cierta cantidad de números impares. Acepta dos parámetros:

- el primero (obligatorio) es el número de impares que devolverá
- el segundo indicará cuál es el primer número impar que debe mostrar. Si se omite este argumento asumirá que es a partir del primer número impar.

Tu función debe recoger el error que se producirá si el segundo parámetro no es un número impar.

```
class ImparesError (¿?):
    # Escribe el código aquí.

def  impares (??);
    # Escribe el código aquí.

# Genera varias series de números impares
# Recoge el tratamiento de posibles excepciones que se puedan generar
```

## 2. TABLA_MULTIPLICAR.PY

Diseña un generador que devuelva la tabla de multiplicar del número que se le pasa como parámetro.

## 3. POTENCIAS_2.PY

Diseña un generador que devuelva las n primeras potencias de dos de un número recogido como argumento.

## 4. FIBONACCI.PY

Diseña un generador que devuelva los n primeros números de la serie de Fibonacci, siendo n el número que se le pasa como parámetro.

Por recordar, la serie de Fibonacci es: 1 - 1- 2 - 3 - 5 - 8 - 13 - 21 - 34 . Los dos primeros número son 1, y el resto se obtiene sumando los dos números anteriores

## 5. MAP

Dada una lista de números enteros, crea una nueva lista con el **triple** de cada número.

Utiliza para realizar el ejercicio las funciones ***list*** y ***map***.

La función map aplica una determinada función a cada uno de los elementos de una entrada o lista.

```
map(funcion_a_aplicar, lista_de_entradas)
```

La función list crea una lista a partir de los valores que se le pasan como parámetros.

```
list(map (……, ……))  crea una lista con los valores obtenidos después de "mapear"
```

- En una primera versión utiliza el método incrementa (debes codificarlo tú) como función_a_aplicar en la función map
- En una segunda versión utiliza una función lambda como función_a_aplicar en la función map

## 6. *FILTER*

A partir de una lista de números, obtén otra lista que contenga solamente los números pares que sean mayores de 10.

Utiliza para realizar el ejercicio las funciones ***list*** y ***filter***.

La función filter devuelve los elementos de una entrada o lista que cumplen una determinada condición. La condición se suele escribir como una función que devuelve True o False, según el elemento de la entrada cumpla o no la condición.

```
filter(filtro_a_aplicar, lista_de_entradas)
```

La función list crea una lista a partir de los valores que se le pasan como parámetros.

```
list(filter (……, ……))  crea una lista con los valores obtenidos después de "filtrar"
```

- En una primera versión utiliza el método condicion (debes codificarlo tú) como filtro_a_aplicar en la función filter
- En una segunda versión utiliza una función lambda como iltro_a_aplicar en la función filter
- En una tercera versión utiliza la compresión de listas.

## 7. *SORTED*

Dada una lista de tuplas (nombre, edad), ordénalas de mayor a menor edad.

Utiliza para realizar el ejercicio la función ***sorted***()

La función sorted() devuelve una nueva lista de elementos, obtenida a partir del iterable que recibe como parámetro. Sorted recorre el iterable elemento a elemento y los va comparando. Como segundo parámetro puede aceptar la key de ordenación, que consiste en la función que se aplicará a cada uno de los elementos para obtener el valor que se usará como criterio de ordenación

En este caso, sorted(listaPersonas) ordenará por el primer valor de cada tupla. Para que ordene según la edad, habrá que especificar la función que aplicada a cada elemento de la lista (o sea, a cada tupla) devuelva el valor de la edad

```
sorted(listaPersonas, key=Obtener-edad)
```

- En una primera versión utiliza el método obtener-edad (debes codificarlo tú) como key de ordenación
- En una segunda versión utiliza una función lambda como key de ordenación

## 8. *FILTER Y DICCIONARIOS*

Dado un diccionario con productos y precios, obtén una lista con los productos que cuestan más de 50, en la forma (producto, precio) .

Utiliza las funciones filter, list,