# Clase 10. Funcional 2

**Ejercicio 1**

Utilizando listas por comprensión, generar una lista de los primeros 10 números naturales elevados al cuadrado

```
print([x * x for x in range(0, 11)])
```

**Ejercicio 2**

Utilizando una función lambda, escribir una función que tome como parámetro una oración y retorne una lista con el largo de cada palabra.

```
s = "pepe que tal como va"

print(list(map(lambda s: len(s), s.split(" "))))
```

**Ejercicio 3**

Dar el resultado de sumar todos los números primos del 1 al 1000

```
tope=1001
primes = [num for num in range(tope) if 0 not in [num%i for i in range(2,num)]]

print(functools.reduce(lambda x, y: x+y, primes))
```

**Ejercicio 4**

Escribir una función (clásica) que reciba cuatro parámetros: lista, n, inc_1, inc_2 y devuela otra lista
- lista es una lista (de largo 10) compuesta por valores numéricos aleatorios entre 1 y 50
- n es un entero, todos los números del parámetro lista mayores o iguales a n deben incrementarse en inc_2, los restantes se incrementarán en inc_1
La lista a devolver se debe resolver usando comprensión, la lista pasada como parámetro también.

```
def inc(lista, n, inc_1, inc_2):
    return [x + inc_2 if x >= n else x + inc_1 for x in lista]
```

**Ejercicio 5**

Obtener una matriz de dos dimensiones a partir de un string del siguiente modo
String: “Hola esto es Python en Antel”
Matriz resultado:
[[“HOLA”, “Hola”, 4] , ]
[“ESTO”, “esto”, 4],
[“ES”, “es”, 2], ….

```
words = 'The quick brown fox jumps over the lazy dog'.split()

stuff = [[w.upper(), w.lower(), len(w)] for w in words]
```

**Ejercicio 6**:

Utilizando una función lambda y la función reduce, escribir una función que tome una lista y devuelva la lista sin repetidos

```
lista = [1, 2, 2, 3, 3, 3, 4, 3]

print(functools.reduce(lambda x, y: x + [y] if not y in x else x, lista, []))
```