Functional Programming
Part 2

1. Map
2. Filter
3. Reduce
4. Zip
5. *args and **kwargs

MONDAY

Optional

→ Applications of Decorator

FP

1) Focuses on immutability
2) looks like mathematical function

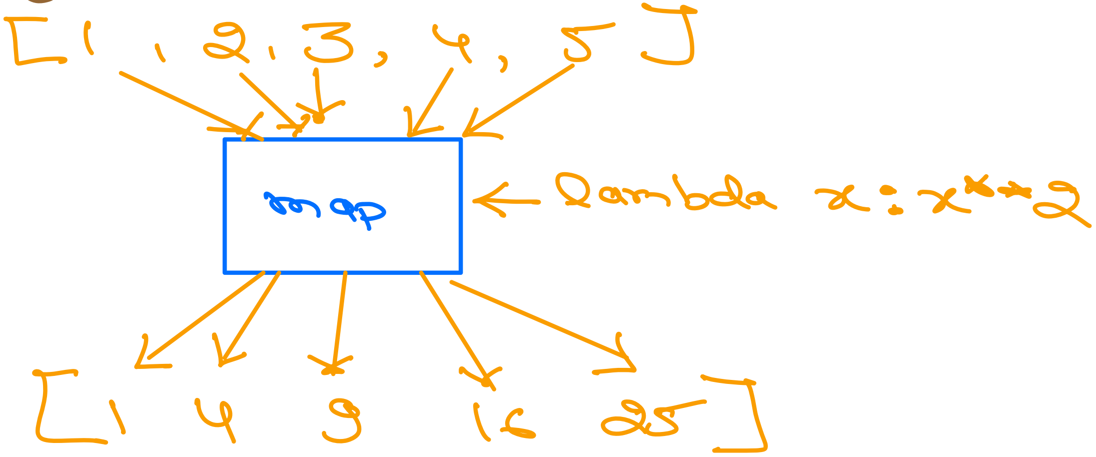
HOF

→ Function that accepts another function as argument or returns function

→ Decorator → Does Both

Map

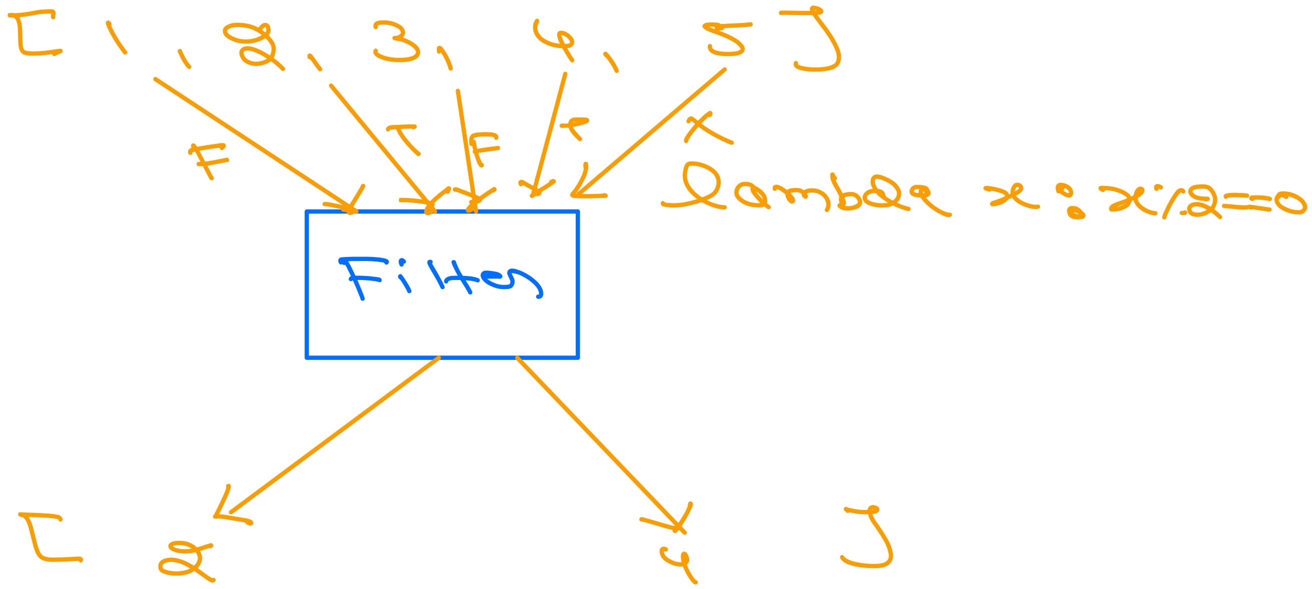
1) Replacement of FOR LOOP
2) Takes an iterable and returns iterable of Same Size
3) Slightly faster



T(map)
T(for loop) > O(n)

Filter

1) Takes function and iterable as input and returns iterable
2) Function should return True or False. (else python converts returned value to Bool)
3) Less than or equal to input size



Zip

1) Zips two or more iterables and returns iterable of Tuples

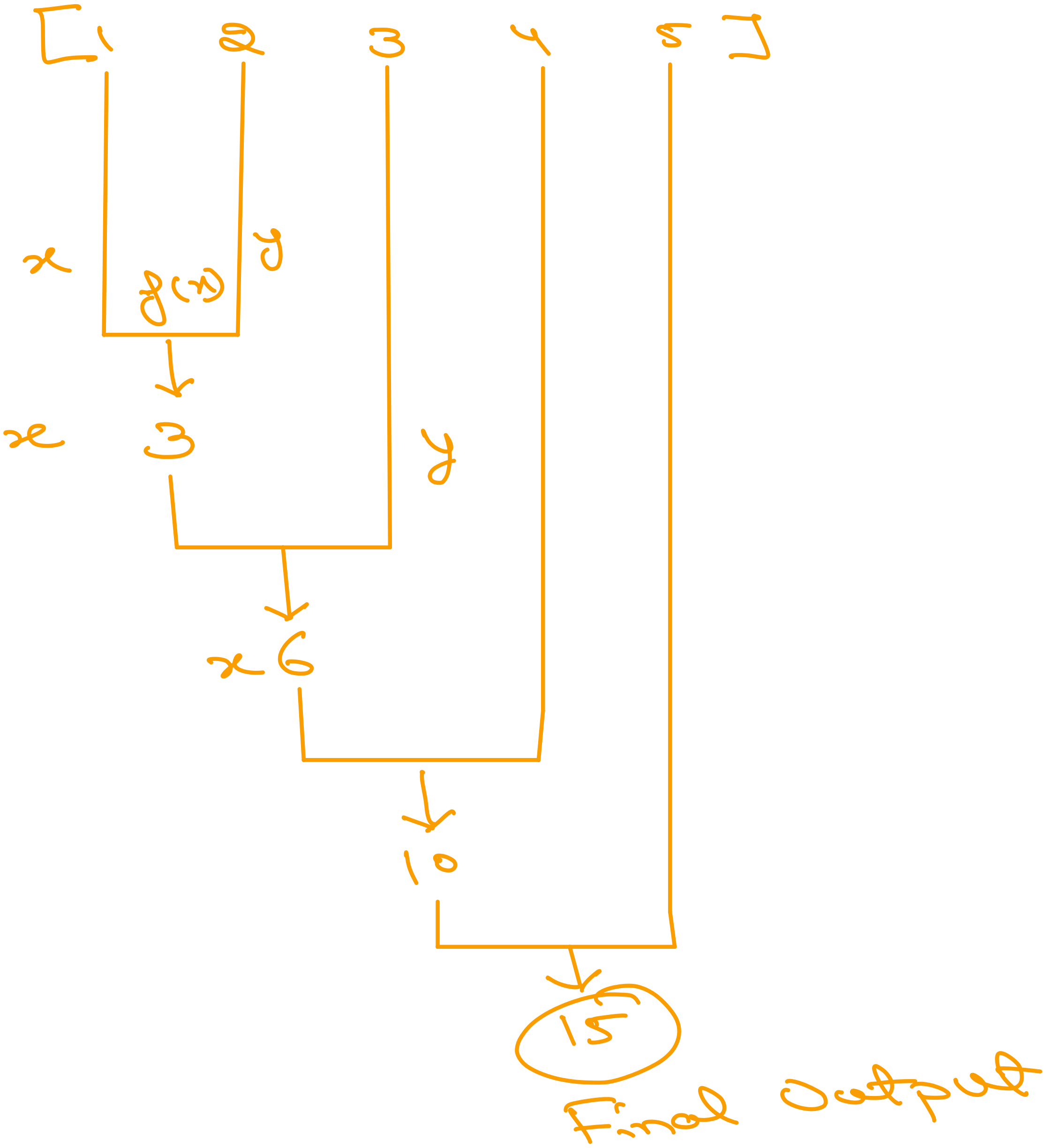
2) The output iterable size depends on min-size of input

list1 => [1, 2, 3, 4]
list2 => [5, 6, 7, 8]

out => list(Zip(list1, list2]
→ [(1, 5), (2, 6), (3, 7), (4, 8)]

Reduce

- Takes function and iterable
- Reduce ← (lambda x, y : x+y)
- Single Value (No generator)


[1 2 3 4 5]
x
y
f(x)
x 3 y
x 6
10
15
Final Output