Functional Programming
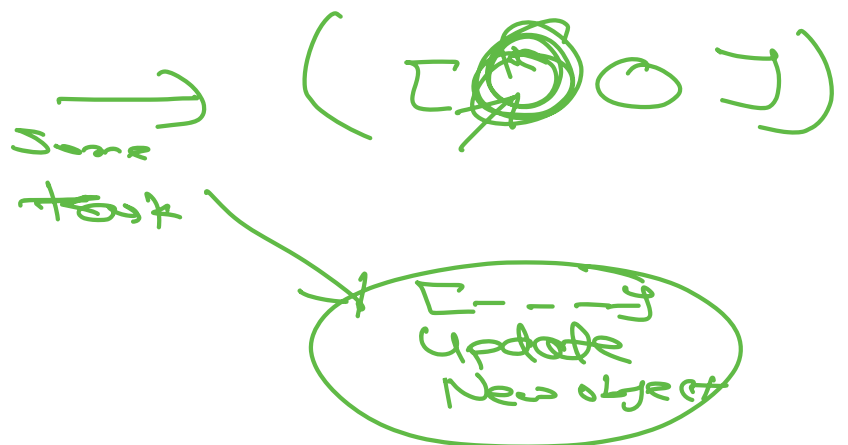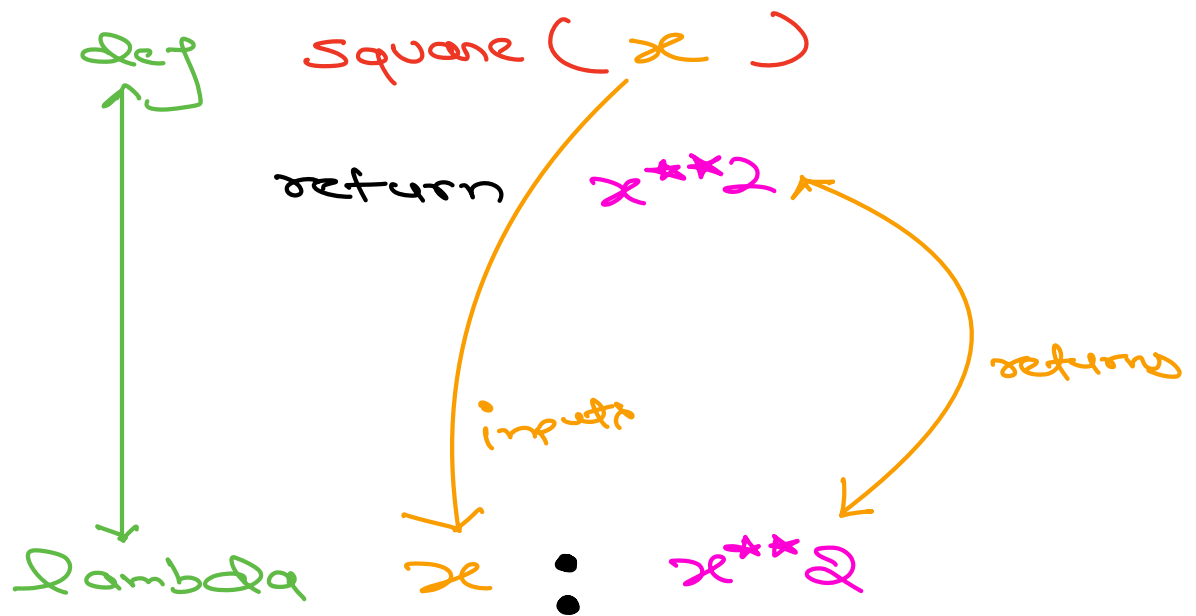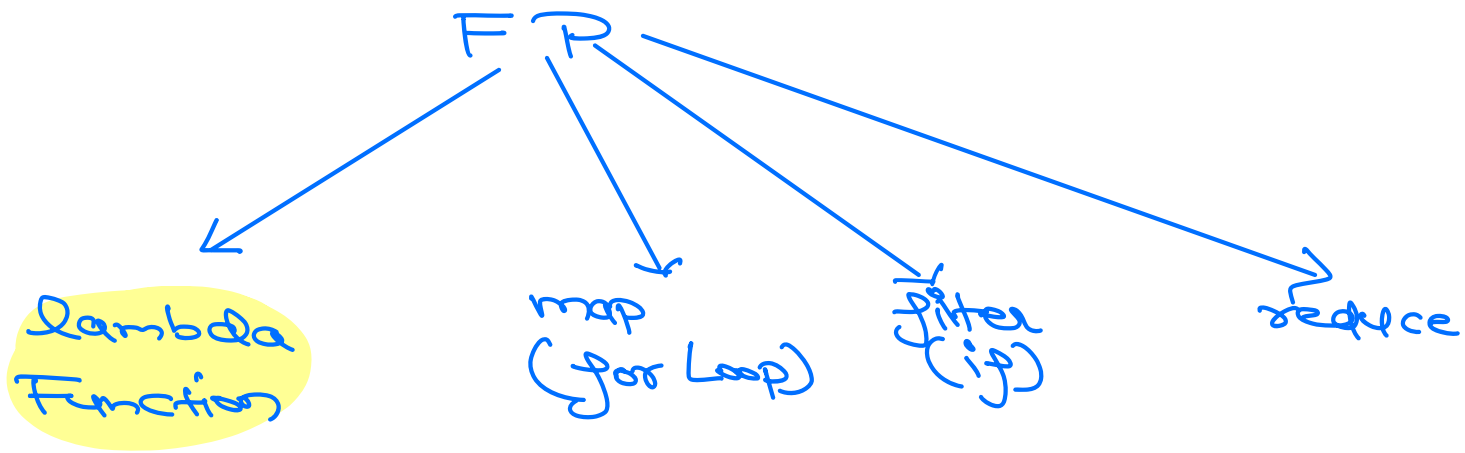
① State change is not allowed

② Declarative: focuses on what the task is

→ FP is more concise and readable *

$$y = f(x)$$

→ More maintainable: Chances of Error due to change in code are less

→ Efficient

→ Same task

[- - -y
Update
New object

FP

lambda
Function

map
(for Loop)

filter
(if)

reduce

def square (x)

return $x{**}2$

lambda

x : $x{**}2$

input

returns

$$y = f(x)$$

* Lambda functions are also known
as Anonymous Function

HOF

Higher Order Functions

① Function that **returns** another Functions

② Used for creating new-variations of Existing Functions On the fly

def **gen exp (n)** : ← ————— HOF

   def exp (x) :

      $x ** n$

   return exp

e-5 = gen- exp (5)

e-5 = $(x ** 5)$ ⟹ exp(x) :

   $x ** 5$

e-r (10) ⟹ $10 ** 5$

   ↓
   x

e_root = gen_exp(0.5)

exp(x)

x ** 0.5

Decorator

⊙ Takes Functions as input

adds more Functionalities

⊙ returns function

FastAPI
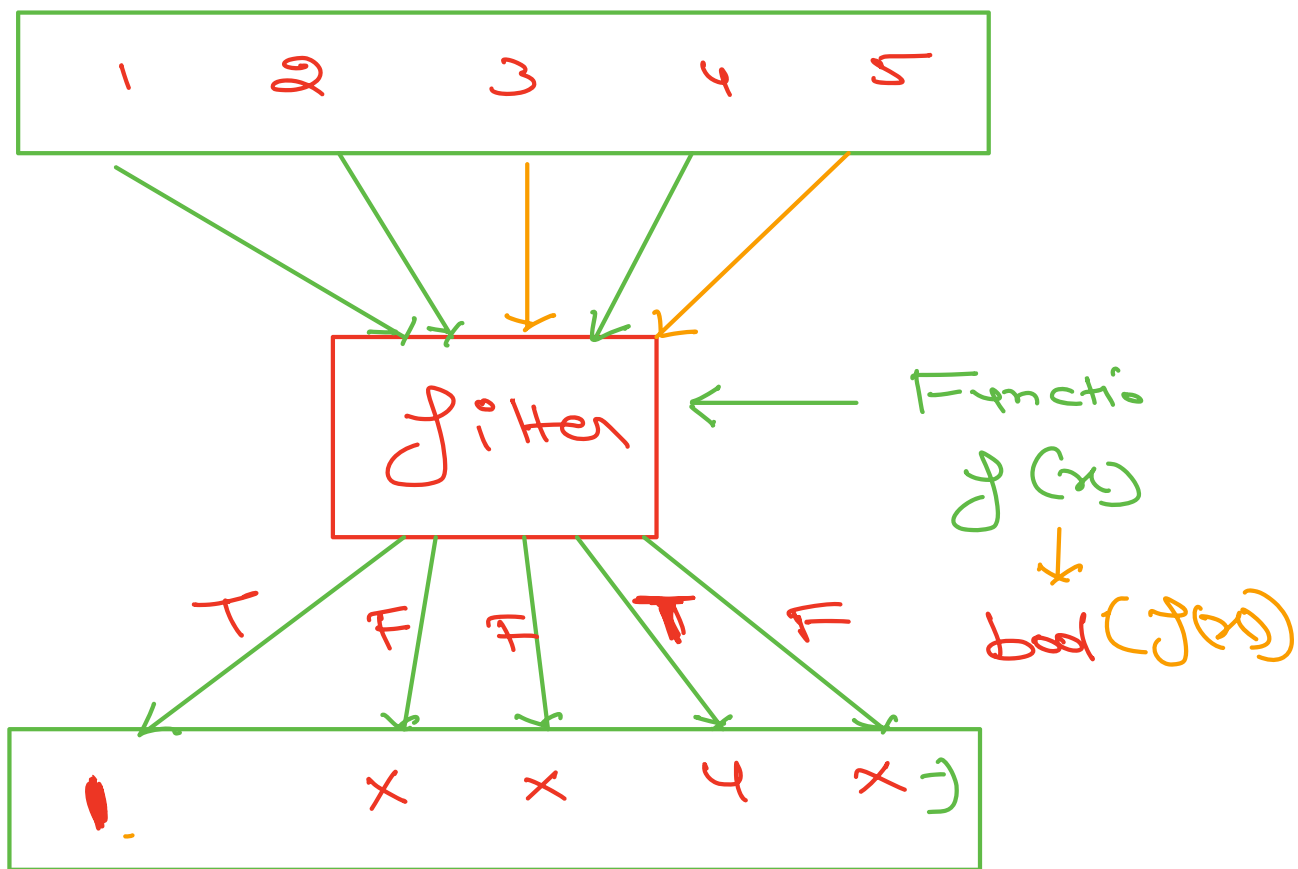FlaskAPI

map

map is an alternative to For loop

| 1 | 2 | 3 | 4 | 5 |

map ← Function $f(x)$

| $f(1)$ | $f(2)$ | $f(3)$ | $f(4)$ | $f(5)$ |
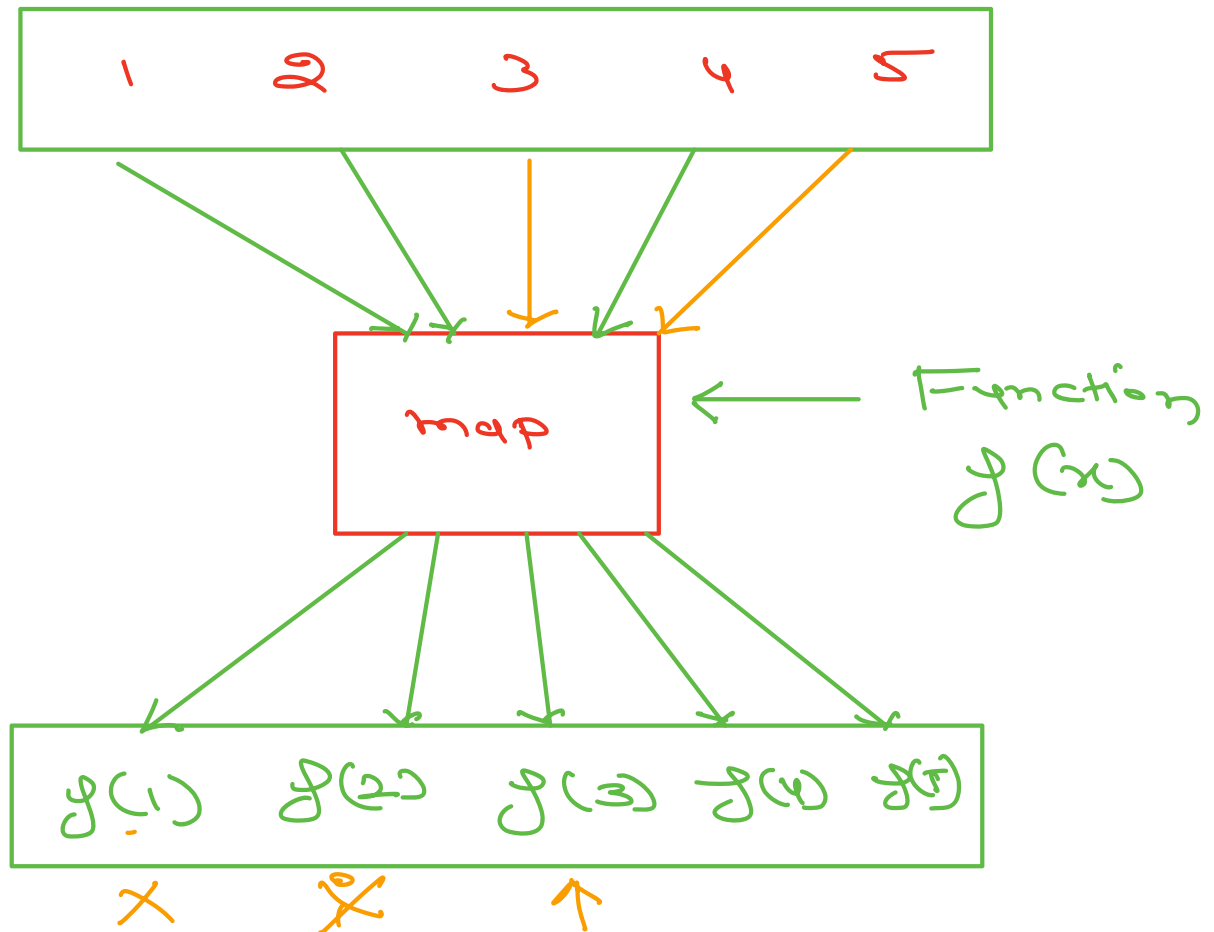
Note: Map returns mapper object

→ iterate

→ Convert list, tuple, set

1   2   3   4   5

filter ← Function $y(x)$

↓

$bool(y(x))$

T   F   F   **T**   F

•   x   x   4   x

Output - length $\leq N$