Functional Programming

- Python supports functional programming
- Builtins for map, reduce, filter, closures, continuations, etc.
- These are often used with lambda

Iterable

- **an object which can be looped over or iterated over with the help of a for loop**. Objects like lists, tuples, sets, dictionaries, strings, etc. are called iterables . In short and simpler terms, iterable is anything that you can loop over.

- We can pass an entire function as a parameter

- Python has functions as first-class citizens

- You simply pass the functions by name

Higher Order Function

- A higher-order function is a function that takes another function as a parameter
- “higher-order” function - Represented as function of a function
- Examples
  - Map
  - Reduce
  - Filter
- Lambda works great as a parameter to higher-order functions if you can deal with its limitations

map(function, iterable, ...)

- Map applies **function** to each element of **iterable** and creates a list of the results
- You can optionally provide more iterables as parameters to map and it will place tuples in the result list
- Map returns an iterator which can be cast to list

Example

```
n = [0, 4, 7, 2, 1, 0 , 9 , 3, 5, 6, 8, 0, 3]

n = list(map(lambda x : x % 7, n))

print(n)
```

Output:

```
[0, 4, 0, 2, 1, 0, 2, 3, 5, 6, 1, 0, 3]
```