itertools

A lightning talk by **Daniel Nephin**

fast, memory

efficient

succinctly

Slicing

```
>>> seq = range(200)
>>> itertools.islice(seq, 5, 10))
5, 6, 7, 8, 9
```

Infinite

```
>>> itertools.count(start=3, step=2)
3, 5, 7, 9, 11, 13, 15, 17, 19, 21, ...

>>> itertools.cycle(range(3))
0, 1, 2, 0, 1, 2, 0, 1, 2, 0, 1, 2, ...

>>> itertools.repeat('a')
'a', 'a', 'a', 'a', 'a', 'a', 'a', ...
```

Filtering

```
>>> eq_to_six = functools.partial(operator.eq, 6)
>>> filter(eq_to_six, range(200))
6

>>> itertools.filterfalse(eq_to_six, range(200))
0, 1, 2, 3, 4, 5, 7, 8, 9, 10, ...

>>> lt_six = functools.partial(operator.gt, 6)
>>> itertools.dropwhile(lt_six, range(200))
6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...

>>> itertools.takewhile(lt_six, range(200))
0, 1, 2, 3, 4, 5
```

Splitting

```
>>> div_by_three = lambda v: v // 3
>>> itertools.groupby(range(10), key=div_by_tree)
(0, [1, 2]), (1, [3, 4, 5]), (2, [6, 7, 8]), ...

>>> itertools.tee(range(4), 3)
(0, 1, 2, 3), (0, 1, 2, 3), (0, 1, 2, 3)
```

Combining

```
>>> itertools.chain(['a', 'b', 'c'], [1, 2, 3])
'a', 'b', 'c', 1, 2, 3

>>> itertools.zip_longest(range(3), ascii_lowercase)
(0, 'a'), (1, 'b'), (2, 'c'), (None, 'd'), ...
```

Combinatorics

```
>>> itertools.product('ABCD', 'xy')
('A', 'x'), ('A', 'y'), ('B', 'x'), ('B', 'y'), ...

>>> itertools.combinations('abc', 2)
('a', 'b'), ('a', 'c'), ('b', 'c')

>>> itertools.permutations('abc')
('a', 'b', 'c'), ('a', 'c', 'b'), ('b', 'a', 'c'),...
```

itertool gems

Period

```
import itertools, operator

period = 200
cycles = map(operator.not_,
             itertools.cycle(range(period)))

# iterate forever
for cycle_is_complete in cycles:
    if cycle_is_complete:
        # Do occasionally
        ...
    # Do this every time
    ...
```

Flatten

```
import itertools

nested_lists = [
    [1, 2, 3],
    [12, 19, 20],
    [20, 40, 50],
]
itertools.chain.from_iterable(nested_lists)
1, 2, 3, 12, 19, 20, 20, 40, 50
```

Segment

```
import itertools, string

def build_groups(size, iterable, fill=None):
    seq = itertools.repeat(iter(iterable), size)
    return itertools.zip_longest(*seq, fillvalue=fill)

build_groups(3, string.ascii_lowercase)
('a', 'b', 'c'), ('d', 'e', 'f'), ('g', 'h', 'i'),...
```

docs.python.org/library/itertools.html