Python Basics

Statements

Markus Dale, 2018

Python - making statements

if, elif, else - statements/ifs.py

```
# first condition evaluating to true gets executed
    if count > 10:
        print('big')
    elif count < 10 and count > 5:
        print('medium')
    else:
        print('small')
```

for loop - break, continue, for/else - statements/fors.py

```python
countries_to_capitols = country_to_caps()

capitols_to_countries = {}

# We want to iterate over all items
for k,v in countries_to_capitols.items():
    capitols_to_countries[v] = k
```

while loop - statements/whiles.py

```
while test:
   statements
   if test: break                   # Exit loop now, skip els
   if test: continue                # Go to test at top of lo
else:
   statements                       # Run if we didn't hit a
```

Source(Mark Lutz, Learning Python 5th edition, O'Reilly, 2013)

Iterations - statements/iterations.py

- for iteration over sequences (e.g. list, sets)
- but also iterables: have `__next__` method and raise StopIteration error at end

```
for c in my_list:
    print(c)

for line in open('three_lines.txt'):
    print(line, end='')
```

List comprehension - statements/iterations.py

```python
# Most general syntax
new_list = [str(elem) for elem in old_list if test]

l = range(100)
# 1. call next on iterator over old_list to get new elem
# 2. check if elem passes the filter's test
# 3. If pass - return elem as part of the new list
# 4. If fail - goto 1
new_l = [num for num in l if num % 2 == 0]
```

Set and dictionary comprehensions - statements/iterations.py

```
letters = ['A', 'c', 'C', 'D', 'D', 'e']
unique_uppers = {letter for letter
                  in letters if letter.isupper()}

lowers = [letter for letter in string.ascii_lowercase]

# ord(c) returns ascii value for c
# chr(num) converts num to character
ascii_vals = [ord(letter) for letter in lowers]

ascii_dict = {letter: ascii for letter,ascii in zip(lowers
```

Functions on iterables - statements/iterations.py

- list, set, dict
- enumerate - creates two-tuple with first element (0, first_iter),
- zip
- map
- sorted
- join
- sum, any, all, max, min