Introduction to Python

What is Python?

- General purpose programming language
- Dynamic, interpreted language
- Emphasizes code readability
- Name is based on a comedy show Monty Python
- Code blocks are defined by indentations
- Code is checked during run time
- Conceived by Guido Van Rossum in 1989

What we will cover

- Running Python
- Data Types
- Control Flow Statements
- Functions
- Classes
- Strings
- Lists and Tuples
- Dictionaries and Files

Hello World!

```
 1 # import statements go here
 2
 3
 4 # define a main function that displays a greeting
 5 def main():
 6     name = input('What is your name? ')
 7     print('Hello, ' + name + '!')
 8
 9
10 if __name__ == '__main__':
11     main()
```

Running Python

Interactive:

```
$ python3
```

From file:

```
$ python3 boilerplate.py
```

Data types

Integers

```
>> num = 4
```

Floats

```
>> num = 4.6
```

Boolean

```
>> True
>> False
```

Strings

```
>> s = 'Hello!'
```

Data types

Lists

```
>> seq = [1, 2, 'spam']
```

Tuples

```
>> seq = ('bacon', 3, 6.3)
```

Dictionaries

```
>> dt = {'a': 'Apple', 'b': True, 'c': 2}
```

Control Flow – If Conditionals

```
if condition1:
    statement1
    …
    statementn
elif condition2:
    statementn
else:
    statementn
```

Control Flow – For Statements

```
for item in list:
    statement1
    statementn
```

Control Flow – While Statements

```
while condition:
    statement1
    statementn
```

Controlling the flow

- **break** – exit the loop immediately
- **continue** – skip to the next iteration
- **pass** – do nothing

Functions

```
def name([arg1, arg2, ...]):
    statement1
    statementn
    [return [expression]]
```

Classes

```
class Name(object):
    def __init__([arg1, arg2, ...]):
        statementn
        [return [expression]]
```

Strings

```
>> st = 'hey there'
>> st = “hey there”
>> st = “””
.. hey
.. there
.. “””””””
..
>>
```

Strings - Methods

- s.lower(), s.upper()
- s.isalpha(), s.isdigit(), s.isspace()
- s.startswith(other_s), s.endswith(other_s)
- s.find(other_s)
- s.replace(old, new)
- s.split(delim)
- s.join(list)

Slicing Sequences

```
>> s = 'hello'
>> s[1:4]
'ell'
```

`seq [ i : j ]`

chars starting at index ***i*** and extending up to but not including index ***j***

Slicing Sequences

Negative index refers to the last item in the sequence:

```
>> s[-1]
'o'
```

This applies to all sequence types: *strings, lists, and tuples*

Exercise 1

Objectives:

- string1.py
- string2.py

Lists

```
>> seq = ['a', 'b', 'c']
>> seq[0]
>> seq[:2]
```

List - Methods

- l.append(item)
- l.extend(list2)
- l.insert(index, item)
- l.remove(item)
- l.pop([index])
- l.index(item)
- l.count(item)
- l.sort()
- l.reverse()

List build up

```
seq = []
for i in range(100):
    seq.append(i)
```

Exercise 2

Objectives:

- list1.py (skip the ones that use sorting)

List Sorting

- ***sorted(list, [key=, reverse=])*** – Returns a new sorted list.

Custom List Sorting

```
>> seq = ['aa', 'bbb', 'c']
>> sorted(seq, key=len)
['c', 'aa', 'bbb']
>> sorted(seq, key=len, reverse=True)
['bbb', 'aa', 'c']
```

List Comprehension

```
>> seq = ['aa', 'bbb', 'c']
>> [i + 'x' for i in seq]
['aax', 'bbbx', 'cx']
```

Tuples

```
>> seq = (4, 3)
>> seq[0] = 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: 'tuple' object does not support item
assignment
>> x, y = (5, 23)
```

Sequence Operations

- x in s
- x not in s
- s + t
- s[i]
- s[i:j]
- len(s)
- max(s)
- min(s)

Exercise 3

Objectives:

- list1.py (Ones that use sorting)
- list2.py

Dictionaries

```
>> data = {'id': 3, 'name': 'Monty', 'coord':
(5, 2)}
>> data['name']
>> del data['id']
```

Dictionary Operations

- len(d)
- d[k]
- d[k] = x
- del d[k]
- k in d
- d.items()
- d.keys()
- d.values()

Files

```
>> fh = open('file.txt', 'r')
>> contents = fh.read()
>> fh.close()
```

File Modes

- 'r' – reading
- 'w' – writing
- 'a' – appending
- 'U' – Normalizes line endings to '\n'. Used in conjuction with the other modes (i.e. 'rU')

File Methods

- f.close()
- f.read() - Reads the file contents to a single string
- f.readlines – Reads the file into a list
- f.seek(i) – Moves the file cursor to position ***i***

Reading large Files

```
fh = open('file.txt', 'rU')
for line in fh:
    print(line)
fh.close()
```

Exercise 4

Objectives:

- wordcount.py