Object Oriented Programming in Python 3

Python 3

User-defined Objects

- Python lets us create our own data types
  And define their behavior
- We create a new type by defining a template for the type called a *class*
- The template can then be used to create objects of that type

__init__()
arguments

```
class Point2D():

    def __init__(self, strcoord='0.0,0.0'):
        self.x, self.y = map(float, strcoord.split(','))

    def printxy(self):
        print('x coord:',self.x)
        print('y coord:',self.y)

ainst = Point2D('-200,900')
binst = Point2D()
```

Python Types

- In Python: *Everything is an object*
- Each type's behaviour is defined in its class

  For example:
  - How they behave when used as operands for operators like `+` `*` etc.
  - Or when used as the sequence/iterator in a `for` clause
  - How elements are returned when using the subscript operator `[]`
  - and more
- How this is achieved: by defining *methods*

Special methods

- We have seen the special method `__init__()`
- By defining the `__init__()` method, we define an interface to our object
- Since Python knows this interface, we are providing a way for Python to communicate with our object
- There are several special methods we can define
  Each with a particular name beginning and ending with double underscores (see Python 3 Data Model for a list)
  - Each provides an interface for different components of the Python language to communicate with an object

Special methods

- Next examples illustrate
  - String conversion
  - Subscription

Classes

String conversion

```
class Point2D():

    def __init__(self, x=0.0, y=0.0):
        self.x = x
        self.y = y

    def printxy(self):
        print('x coord:',self.x)
        print('y coord:',self.y)

ainst = Point2D()
print(ainst)
```

```
$ python3 point2d.py
<__main__.Point2D object at 0x7f87097447f0>
```

String conversion

- Before `print()` can output the representation of the object to `stdout`, it needs to convert it to a string
- String conversion can be done by passing the object as an argument to the `str()` built-in function which returns a string
  print() does this implicitly
- Documentation says that calling `str(object)` itself calls `object.__str__()` method and returns the value returned

Special Methods

String conversion

```
class Point2D():

    def __init__(self, x=0.0, y=0.0):
        self.x = x
        self.y = y

    def __str__(self):
        return '{},{}'.format(self.x,self.y)

ainst = Point2D(2.2,-3.3)
print(ainst)
```

```
$ python3 point2d_noxy.py
2.2,-3.3
```

Special Methods

Subscription

- Is when you access an element of a sequence using subscript syntax
  `seq[2]`
- Python interacts with an object that is being subscripted through the `__getitem__()` special method
- `seq[2]` equivalent to `seq.__getitem__(2)`

Special Methods

```
class FibGen:
    def fibn(self, n):
        a, b = 0, 1
        for i in range(n):
            a, b = b, a+b
        return b

    def __getitem__(self, n):
        return self.fibn(n)
```

```
>>>> import fibclass as f
>>>> fg = f.FibGen()

>>>> print(fg[8])
34

>>>> print(fg[31], fg[32], fg[33])
2178309 3524578 5702887
```

Extended Example

A *StopWatch* class

- While programming, we may need to determine how long a set of instructions or a function call took to complete
  - A form of performance analysis
- One method is to record the time before and after the code/function in question
- We develop a new type called StopWatch that can perform this function in an object oriented way

Extended Example

A *StopWatch* class

- Imagine we wanted to compare the time taken to sort a list by 1) A `bubblesort()` function that we define and 2) Python's built-in `sorted()` function
- We want to be able to do this

```
sw = StopWatch()
sw.start()
bubblesort(mylist)
sw.stop()
print(sw)
0:0:1.0011961460113525
```

Extended Example

StopWatch class interface

- `start()`
- `stop()`
- `split()`
- `reset()`
- `gettime()`

In addition

- Should return meaningful output when passed to print function
- Should be subscriptable to access split times
- + operator should be defined

Homework