Quiz 1

A. Structuring Techniques

Modules and classes: what is the difference?

A module is essentially just a file containing Python code. It may contain global variables, functions, classes. Modules cannot be nested. You do not make instances of modules. You import modules.

Modules cannot be nested. (You cannot declare a new module inside a module)

Classes can be nested. (You can declare a class inside a class.)

Functions can be nested.

Functions versus methods. A method is a function inside a class.
All methods are functions. But not all functions are methods.

Two kinds:

- instance methods: have the keyword self as first parameter
- class methods: do not have the keyword self as the first parameter

These are "helper" functions, and you have to put the class name DOT in front

Here's an interesting twist: in Python, functions are objects! They can be passed around.

B. Function parameters

Lots about functions and parameters. Here are two tidbits:

0. Normally functions get their parameter values from the call statement by position.

1. default parameter values: simply put a value after parm= and if you omit it, Python will use that instead.

```python
def pay(amount, name="Jane", when="tomorrow"):  
  pass  

pay(825, "Tony")
```

So you can call:

```python
pay(100.57) and it will pay that amount to Jane tomorrow
```

or you can call:

```python
pay(500.22, "Mark", "yesterday")
```
and override the defaults.

Obviously, the parameters with defaults must come after the ones by position.

2. names parameter values: use the names of the parameters, in which case you can mix them up. But if there is no default value, you must supply it.

Using the pay function above, you can do
\[
\text{pay(name="Sally", amount=843.00, when="in 2 days")}
\]

If you put the amount first, then the others can be called by name:
\[
\text{pay(206.00, when="next week", name="Bob")}
\]

4. There are even more things you can do with function parameters but this is all you will need.

C. Object Oriented Programming Review

Review: How to define a class
   How to make an instance of a class (i.e. an object)

Some examples:

Fractions:

Complex numbers

Point class: what are its members and methods? (pretty minimal, could convert to polar)

Line: what are its members? what are its methods? (slope, distance between points)
   is the line finite or infinite? Defined by two points, or by a point and a slope?

RWOs = Real World Objects
   Examples: book, library, employee, department

   Clocks and calendars
      clocks—military time (24-hour)
      12-hour clock

Containers:
   list, set, tuple, bag, limited bag, ordered set

List: ordered, heterogeneous, not unique
Set: unordered, hetero-, unique
Bag: unordered, hetero-, not unique
Limited Bag: like a Bag but there are limits on the number of instances (applies to all or a different limit for each value?)
Ordered Set: ordered, hetero-, unique
class Cat:
    """Model an instance of Felis catus""
    def __init__(self, name, age):
        self.name = name
        self.age = age
        self.breed = "Savannah"  # "DSH"
        self.hungry = True
    def feed(self, howmuch):
        """howmuch is ounces of dry cat food""
        self.hungry = False
kitty = Cat("Luke", 5)
kitty.breed = "DSh"