14.8: MANY INSTANCES

Contributed by Chuck Severance
Clinical Associate Professor (School of Information) at University of Michigan

So far, we have been defining a class, making a single object, using that object, and then throwing the object away. But the real power in object oriented happens when we make many instances of our class.

When we are making multiple objects from our class, we might want to set up different initial values for each of the objects. We can pass data into the constructors to give each object a different initial value:

CODE 14.8.1 (PYTHON):

```
%%python3

class PartyAnimal:
   x = 0
   name = ''
   def __init__(self, nam):
     self.name = nam
     print(self.name,'constructed')

   def party(self) :
     self.x = self.x + 1
     print(self.name,'party count',self.x)

s = PartyAnimal('Sally')
j = PartyAnimal('Jim')

s.party()
j.party()
s.party()

# Code: http://www.py4e.com/code3/party5.py
```

run restart

The constructor has both a `self` parameter that points to the object instance and then additional parameters that are passed into the constructor as the object is being constructed:

```
s = PartyAnimal('Sally')
```

Within the constructor, the line:

```
self.name = nam
```

Copies the parameter that is passed in (`nam`) into the `name` attribute within the object instance.

The output of the program shows that each of the objects (`s` and `j`) contain their own independent copies of `x` and `nam` :

```
Sally constructed
Sally party count 1
Jim constructed
Jim party count 1
Sally party count 2
```