0.1 Python Classes

This tutorial seeks to remind us about basics, and some fine points, related to object oriented mechanisms in Python:

- instance and class attributes
- inheritance

0.1.1 Instance and Class Attributes

In [ ]: class A:
   
x = "cat"

   a = A()
   print("a.x:", a.x)

In [ ]: x = "dog"

   class B(A):
      
x = "ferret"
      
def __init__(self):
         self.x = 'tomato'

   b = B()
   print("b.x:", b.x)

   # What will be printed?
   # 1 - b.x: cat
   # 2 - b.x: dog
   # 3 - b.x: ferret
   # 4 - b.x: tomato
   # 5 - Other or error

In [ ]: class A:
   
x = "cat"

   class B(A):
      
x = "dog"
      
def __init__(self):
self.x = "ferret"

class C(B):
    x = "tomato"

c = C()
print("c.x:", c.x)

# What will be printed?
# 1 - c.x: cat
# 2 - c.x: dog
# 3 - c.x: ferret
# 4 - c.x: tomato
# 5 - Other or error

0.1.2 Method inheritance

Methods can be thought of as class (or instance) attributes that happen to be functions: they are resolved similarly, then called on their arguments. There are various special syntactic forms and protocols that govern instance creation, initialization, destruction, as well as method invocation syntax to make it convenient to pass along the instance object itself.

In [ ]: class Bar():
    def __init__(self, val):
        self.x = val

class Foo(Bar):
    x = 100
    def increment(this):
        # conventionally 'self' rather than 'this' or other variable
        this.x += 1

f = Foo(33)
print("f.x:", f.x)
f.increment()
print("f.x:", f.x)

Invoking a superclass method:

In [ ]: class Bar():
    def __init__(self, val):
        self.x = val

class Foo(Bar):
    x = 100
    def __init__(self, val):
        Bar.__init__(self, val)
        #super().__init__(val)
        self.x = self.x * Foo.x
def increment(self):
    self.x += 1

f = Foo(33)
print("f.x:", f.x)
f.increment()
print("f.x:", f.x)

Invoking a subclass method from a superclass:

In [ ]: class Bar():
    def __init__(self, val):
        self.x = val

    def double_increment(self):
        self.increment()
        self.increment()

class Foo(Bar):
    def increment(self):
        self.x += 1

class Gorp(Bar):
    delta = 100
    def increment(self):
        self.x += self.delta

f = Foo(0)
print("f.x:", f.x)
f.double_increment()
print("f.x:", f.x)

g = Gorp(0)
print("g.x:", g.x)
g.double_increment()
print("g.x:", g.x)