To test a user-defined class, you will create test cases that check whether instances are created properly, and you will create test cases for each of the methods as functions, by invoking them on particular instances and seeing whether they produce the correct return values and side effects, especially side effects that change data stored in the instance variables. To illustrate, we will use the Point class that was used in the introduction to classes.

To test whether the class constructor (the \_\_init\_\_) method is working correctly, create an instance and then make tests to see whether its instance variables are set correctly. Note that this is a side effect test: the constructor method’s job is to set instance variables, which is a side effect. Its return value doesn’t matter.

A method like distanceFromOrigin in the Point class you saw does its work by computing a return value, so it needs to be tested with a return value test. A method like move in the Turtle class does its work by changing the contents of a mutable object (the point instance has its instance variable changed) so it needs to be tested with a side effect test.

Try adding some more tests in the code below, once you understand what’s there.

**CODE**

class Point:

""" Point class for representing and manipulating x,y coordinates. """

def \_\_init\_\_(self, initX, initY):

self.x = initX

self.y = initY

def distanceFromOrigin(self):

return ((self.x \*\* 2) + (self.y \*\* 2)) \*\* 0.5

def move(self, dx, dy):

self.x = self.x + dx

self.y = self.y + dy

#testing class constructor (\_\_init\_\_ method)

p = Point(3, 4)

assert p.y == 4

assert p.x == 3

#testing the distance method

p = Point(3, 4)

assert p.distanceFromOrigin() == 5.0

#testing the move method

p = Point(3, 4)

p.move(-2, 3)

assert p.x == 1

assert p.y == 7