20.6. Objects as Arguments and Parameters

You can pass an object as an argument to a function, in the usual way.

Here is a simple function called `distance` involving our new `Point` objects. The job of this function is to figure out the distance between two points.

Save & Run Original - 1 of 1 Show in CodeLens

```
import math

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

    def __init__(self, initX, initY):

        self.x = initX
        self.y = initY

    def getX(self):
        return self.x

    def getY(self):
        return self.y
```

```
5.0
```

Activity: 1 -- ActiveCode (chp13_classes6)

`distance` takes two points and returns the distance between them. Note that `distance` is **not** a method of the Point class. You can see this by looking at the indentation pattern. It is not inside the class definition. The other way we can know that `distance` is not a method of Point is that `self` is not included as a formal parameter. In addition, we do not invoke `distance` using the dot notation.

We *could have* made distance be a method of the Point class. Then, we would have called the first parameter self, and would have invoked it using the dot notation, as in the following code. Which way to implement it is a matter of coding style. Both work correctly. Most programmers choose whether to make functions be stand-alone or methods of a class based on whether the function semantically seems to be an operation that is performed on instances of the class. In this case, because distance is really a property of a pair of points and is symmetric (the distance from a to b is the same as that from b to a) it makes more sense to have it be a standalone function and not a method. Many heated discussions have occurred between programmers about such style decisions.

Save & Run Original - 1 of 1 Show in CodeLens

```
import math

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

    def __init__(self, initX, initY):

        self.x = initX
        self.y = initY

    def getX(self):
        return self.x

    def getY(self):
        return self.y
```

```
5.0
```

Activity: 2 -- ActiveCode (chp13_classes6a)

You have attempted 3 of 2 activities on this page

20.5. Adding Other Methods to a Class">
20.7. Converting an Object to a String">
20.5. Adding Other Methods to a Class">
Completed. Well Done!
20.7. Converting an Object to a String">Next Section - 20.7. Converting an Object to a String

| Back to top