Chapter 7 :
Objects and Classes

Aviral Janveja

1 Object

Throughout this course in Python, we have seen various examples of data like numbers, booleans, strings, sequences and so on. All of these instances of data belong to a certain type like integer, float, string or list.

Now, object basically refers to data in Python. So no matter whether it is integer data, float data or string data; it is all data in the first place. Hence, data or object itself, is the root of all data types. This base object type provides the most basic and generally assumed background functionality, such as the ability to assign values to variables.

Therefore, it is commonly said that **everything is an object** in Python and every object has a type. Further, the type that a particular object belongs to, defines how it is represented behind the scenes in Python language and also the ways in which one can interact with that object. For example, a list is always defined using square brackets [ ] and you can manipulate a list object using the various methods associated with it like insert, append, remove and sort.

2 Class

The next question that naturally arises is : how are these data-types defined and can we define our own type ?

Answer : Yes, we can define our own data-type by defining a new **class**. Defining your own class is like defining a blue print for your own data-type, similar to integers, floats or lists. The class definition looks like this :

```
class class_name(object):
    #define attributes here
```

We start with the **class** keyword, then we mention the **name** of our class and in the parenthesis we mention the **parent** class. The parent class here is ofcourse object. As we have already discussed, it is the root-parent of all types in Python. Inside the class, we define its attributes which are the data attributes and procedural attributes that belong to the class.

2.1 Data

Data attributes define the data representation of this particular type. For example, a two dimensional coordinate object will be made up of two numbers, in a bracket, separated by a comma.

Coordinate object = (3,5)

One value for the x-coordinate and one value for the y-coordinate. We can further decide on whether these two numbers will be of type int or type float.

2.2 Methods

Procedural attributes, better known as methods are functions, that only work with this particular type. For example, you can define a distance method between two coordinate objects, but that method will have no meaning for list objects.

3 Defining a new type : Class Coordinate

3.1 Special method __init__

special method, used to initialize data attributes.

3.2 Creating objects of type coordinate

3.3 Defining Distance Method