Everything is an object in Python

Python is often looked upon as a scripting language. No. It's not just a scripting language. It's an object oriented general purpose programming language. Though it can very well be used efficiently just for scripting.

And in Python everything is an object.

For a beginner in python like me, the above statement is an oxymoron... How could a scripting language be a general purpose programming language in which everything is an object? Well, after exploring a bit about the language, it's not difficult to understand how the python is made to look like a full blown scripting language. Under the hoods, it is all objects. And to the programmer, it is an elegant and easy to use language.

Let us start with this base heuristic that "Everything is an object in Python" and explore it a little further.

Well, we are used to starting with the "Hello World" example isn't it. Our first example is also the program that prints "Hello World" on the console.

In Python, to print "Hello World", all you need is one line of code

```
print ("Hello World")
```

Typing the above line in the interactive console prints the string "Hello World" in the next line.

```
>>> print ("Hello World")
Hello World
```

Alternately, you can type in this one line in a file with extension .py and execute this file with python command

```
# hello_world.py
print ("Hello World")
```

Executing this yields the following output

```
/python_musings/object_orientation>python hello_world.py
Hello World
```

Now back to our original heuristic that "Everything is an object in Python"

So what is object oriented about our hello world program? Nothing in that one line resembles anything close to objects. On the face of it, it is just one statement in which we are asking Python to print "Hello World"

Let's dig a bit deeper to understand where object comes into picture in our example. Our program just contains two parts

1. print – a built in function in Python. A function is an object in python, to be precise they are callable objects
2. "Hello World" – is a string that we pass as parameter to the print function. And a string is an object in Python

As you can see, our simple hello world program which consists of one line of code with two words in it and both the words turn out to be objects.

Objects are instances of a class. The type of an object is the class which it is an instance of.

Let's try to check the types of "print" and "Hello World". To find that we can use a built in function called type. This returns the type of anything (read object) passed as input to it. And here are the types of "print" and "Hello World"

```
>>> type (print)
<class 'builtin_function_or_method'>
>>> type ("Hello World")
<class 'str'>
```

And as we explore deeper, we find that everything in python turns out to be an object.

Some examples below

```
>>> type (10)
<class 'int'>
>>> type (0.1)
<class 'float'>
>>> i = 1000
>>> type(i)
<class 'int'>
>>> j = i
>>> type (j)
<class 'int'>
>>> type ([1,2,3])
<class 'list'>
>>> type (True)
<class 'bool'>
>>> type(None)
<class 'NoneType'>
```

As we can see, "None" that represents null or nothing is also an object in Python.

And what would be the type of undefined variables? Let's try to find out

```
>>> type(n)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'n' is not defined
```

The type function reports NameError with the message that ‘n’ has not yet been defined. And it reports correct type after defining ‘n’

```
>>> n = "I am defined now"
>>> type(n)
<class 'str'>
```

```
>>> n = 10
>>> type(n)
<class 'int'>
>>> n = 10.1
>>> type(n)
<class 'float'>
>>> n = (1.1, "Hello", 1000)
>>> type(n)
<class 'tuple'>
>>> n = ["one", 2, 3.0]
>>> type(n)
<class 'list'>
>>> n = {"name":"n", "value":"any"}
>>> type(n)
<class 'dict'>
```

As you can see, n can be assigned any type of object and type reports the class that it is an instance of, based on the object assigned to it when type() is invoked.

This brings us to our next question. Do variables have a type? We will explore this in the next chapter

---

* All the source files used in the samples are available under the src directory of the current path