**CLASS:**

Compared with other programming languages, Python’s class mechanism adds classes with a minimum of new syntax and semantics. It is a mixture of the class mechanisms found in C++ and Modula-3. Python classes provide all the standard features of Object Oriented Programming: the class inheritance mechanism allows multiple base classes, a derived class can override any methods of its base class or classes, and a method can call the method of a base class with the same name. Objects can contain arbitrary amounts and kinds of data. As is true for modules, classes partake of the dynamic nature of Python: they are created at runtime, and can be modified further after creation.

**OBJECT:**

We saw that the class object could be used to access different attributes.

It can also be used to create new object instances (instantiation) of that class. The procedure to create an object is similar to a [function](https://www.programiz.com/python-programming/function)call.

1. >>> ob = MyClass()

This will create a new instance object named ob. We can access attributes of objects using the object name prefix.

**Attribute:**

**Attributes** are data stored inside a class or instance and represent the state or quality of the class or instance. In short, **attributes** store information about the instance.

**Behaviour:**

The **behavior of an object** is defined by its methods, which are the functions and subroutines defined within the **object** class. Without class methods, a class would simply be a structure. Methods determine what type of functionality a class has, how it modifies its data, and its overall **behavior**