Object Oriented Programming

Class

A Python class is a group of attributes and methods.

What is Attribute ?

Attributes are represented by variable that contains data.

What is Method?

Method performs an action or task. It is similar to function.

How to Create Class

```
class Classname(object) :
    def __init__(self):
        self.variable_name = value
        self.variable_name = 'value'
    def method_name(self):
        Body of Method
```

Method

Attributes

```
class Classname :
    def __init__(self):
        self.variable_name = value
        self.variable_name = 'value'
    def method_name(self):
        Body of Method
```

- class - class keyword is used to create a class
- object - object represents the base class name from where all classes in Python are derived. This class is also derived from object class. This is optional.
- __init__() – This method is used to initialize the variables. This is a special method. We do not call this method explicitly.
- self – self is a variable which refers to current class instance/object.

Rules

- The class name can be any valid identifier.
- It can't be Python reserved word.
- A valid class name starts with a letter, followed by any number of letter, numbers or underscores.
- A class name generally starts with Capital Letter.

How to Create Class

```
class Mobile:
    def __init__(self):
        self.model = ‘RealMe X’
    def show_model (self):
        print(‘Model:’, self.model)
```

How to Create Class

```
class Classname :
    def __init__(self):
            self.variable_name = value
            self.variable_name = 'value'

    def method_name(self):
            Body of Method
                                        Formal Argument

   def method_name(self, f1, f2):
            Body of Method
```

```
                            Formal Argument
class Classname :
    def __init__(self, f1, f2):
            self.variable_name = value
            self.variable_name = 'value'

    def method_name(self):
            Body of Method
                                        Formal Argument

   def method_name(self, f1, f2):
            Body of Method
```

How to Create Class

```
class Mobile:
    def __init__(self, m):
        self.model = m
    def show_model (self, p):
        price = p                    # Local Variable
        print(‘Model:’, self.model, ‘Price:’, price)
```

Object

Object is class type variable or class instance. To use a class, we should create an object to the class.

Instance creation represents allotting memory necessary to store the actual data of the variables.

Each time you create an object of a class a copy of each variables defined in the class is created.

In other words you can say that each object of a class has its own copy of data members defined in the class.

Syntax: -

```
object_name = class_name()
object_name = class_name(arg)
```

How to Create Object

```
class Mobile:
    def __init__(self):
        self.model = ‘RealMe X’
    def show_model (self):
        print(‘Model:’, self.model)

realme = Mobile()
```

```
class Mobile:
    def __init__(self, m):
        self.model = m
    def show_model (self):
        print(‘Model:’, self.model)

realme = Mobile(‘RealMe X’)
```

How it works

realme = Mobile()

- A block of memory is allocated on heap. The size of allocated memory is to be decided from the attributes and methods available in the class (Mobile).
- After allocating memory block, the special method __init__() is called internally. This method stores the initial data into the variables.
- The allocated memory location address of the instance is returned into object (realme).
- The memory location is passed to self.

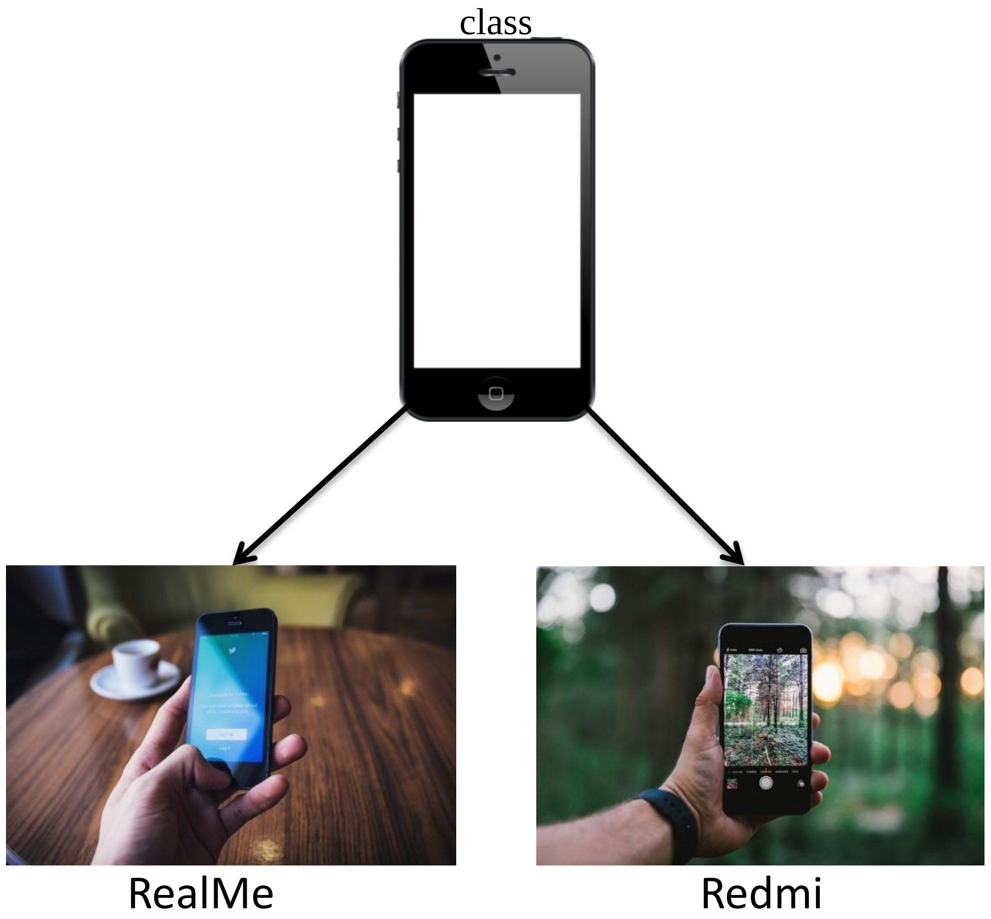
Accessing class member using object

We can access variable and method of a class using class object or instance of class.

```
object_name.variable_name
realme.model

object_name.method_name ( )
realme.show_model ( );

object_name.method_name (parameter_list)
realme.show_model(1000);
```


class
RealMe
Redmi

self Variable

self is a default variable that contains the memory address of the current object.

This variable is used to refer all the instance variable and method.

When we create object of a class, the object name contains the memory location of the object.

This memory location is internally passed to *self*, as *self* knows the memory address of the object so we can access variable and method of object.

self is the first argument to any object method because the first argument is always the object reference. This is automatic, whether you call it *self* or not.

```
def __init__(self):
```

```
def show_model(self):
```

Object

Each time you create an object of a class a copy of each variables defined in the class is created.

```
class Mobile:
    def __init__(self):
        self.model = ‘RealMe X’
    def show_model (self):
        print(‘Model:’, self.model)

realme = Mobile()
redmi = Mobile()
gold = Mobile()
```