# Python's Object Oriented Programming (OOPs)

## What is Class:

- In Python every thing is an object. To create objects we required some Model or Plan or Blue print, which is nothing but class.
- We can write a class to represent properties (attributes) and actions (behaviour) of object.

- Properties can be represented by variables
- Actions can be represented by Methods.

- Hence class contains both variables and methods.

## How to Define a class?

We can define a class by using class keyword.

Syntax:

```
class className:
  ''' documenttation string '''
   variables:instance variables,static and local variables
   methods: instance methods,static methods,class methods
```

Documentation string represents description of the class. Within the class doc string is always optional. We can get doc string by using the following 2 ways.

1. print(classname.__doc__)
2. help(classname)

Within the Python class we can represent data by using variables.
There are 3 types of variables are allowed.

1. Instance Variables (Object Level Variables)
2. Static Variables (Class Level Variables)
3. Local variables (Method Level Variables)

Within the Python class, we can represent operations by using methods. The following are various types of allowed methods

1. Instance Methods
2. Class Methods
3. Static Methods

## What is Object:

Pysical existence of a class is nothing but object. We can create any number of objects for a class.

Syntax to create object: referencevariable = classname()

# What is Reference Variable:

The variable which can be used to refer object is called reference variable.
By using reference variable, we can access properties and methods of object.

# Self variable:

self is the default variable which is always pointing to current object (like this keyword in Java)

By using self we can access instance variables and instance methods of object.

**Note:**
1. self should be first parameter inside constructor
   def __init__(self):

2. self should be first parameter inside instance methods
   def talk(self):

# Constructor Concept:

- Constructor is a special method in python.
- The name of the constructor should be __init__(self)
- Constructor will be executed automatically at the time of object creation.
- The main purpose of constructor is to declare and initialize instance variables.
- Per object constructor will be exeucted only once.
- Constructor can take atleast one argument(atleast self)

- Constructor is optional and if we are not providing any constructor then python will provide default constructor.

# Types of Variables:

Inside Python class 3 types of variables are allowed.

1. Instance Variables (Object Level Variables)
2. Static Variables (Class Level Variables)
3. Local variables (Method Level Variables)

# 1. Instance Variables:

If the value of a variable is varied from object to object, then such type of variables are called instance variables.

For every object a separate copy of instance variables will be created.

## Where we can declare Instance variables:

1. Inside Constructor by using self variable
2. Inside Instance Method by using self variable
3. Outside of the class by using object reference variable

## 1. Inside Constructor by using self variable:

We can declare instance variables inside a constructor by using self keyword. Once we creates object, automatically these variables will be added to the object.

## 2. Inside Instance Method by using self variable:

We can also declare instance variables inside instance method by using self variable. If any instance variable declared inside instance method, that instance variable will be added once we call taht method.

## 3. Outside of the class by using object reference variable:

We can also add instance variables outside of a class to a particular object.

## How to access Instance variables:

We can access instance variables with in the class by using self variable and outside of the class by using object reference.

## How to delete instance variable from the object:

1. Within a class we can delete instance variable as follows

```
del self.variableName
```

2. From outside of class we can delete instance variables as follows

```
del objectreference.variableName
```

## 1. Static variables:

If the value of a variable is not varied from object to object, such type of variables we have to declare with in the class directly but outside of methods. Such type of variables are called Static variables.

For total class only one copy of static variable will be created and shared by all objects of that class.

We can access static variables either by class name or by object reference. But recommended to use class name.

## Various places to declare static variables:

1. In general we can declare within the class directly but from out side of any method
2. Inside constructor by using class name
3. Inside instance method by using class name
4. Inside classmethod by using either class name or cls variable
5. Inside static method by using class name

## How to access static variables:

1. inside constructor: by using either self or classname
2. inside instance method: by using either self or classname
3. inside class method: by using either cls variable or classname
4. inside static method: by using classname
5. From outside of class: by using either object reference or classnmae

## Where we can modify the value of static variable:

Anywhere either with in the class or outside of class we can modify by using classname.
But inside class method, by using cls variable.

## If we change the value of static variable by using either self or object reference variable:

If we change the value of static variable by using either self or object reference variable, then the value of static variable won't be changed,just a new instance variable with that name will be added to that particular object.

## How to delete static variables of a class:

We can delete static variables from anywhere by using the following syntax

```
del   classname.variablename
```

But inside classmethod we can also use cls variable

```
del   cls.variablename
```

## Local variables:

Sometimes to meet temporary requirements of programmer,we can declare variables inside a method directly,such type of variables are called local variable or temporary variables.

Local variables will be created at the time of method execution and destroyed once method completes.

Local variables of a method cannot be accessed from outside of method.

## Types of Methods:

Inside Python class 3 types of methods are allowed

1. Instance Methods
2. Class Methods
3. Static Methods

# 1. Instance Methods:

Inside method implementation if we are using instance variables then such type of methods are called instance methods.
Inside instance method declaration,we have to pass self variable.

```
def m1(self):
```

By using self variable inside method we can able to access instance variables.

Within the class we can call instance method by using self variable and from outside of the class we can call by using object reference.

# Setter and Getter Methods:

We can set and get the values of instance variables by using getter and setter methods.

# Setter Method:

setter methods can be used to set values to the instance variables. setter methods also known as mutator methods.

syntax:

```
def  setVariable(self,variable):
     self.variable=variable
```

# Getter Method:

Getter methods can be used to get values of the instance variables. Getter methods also known as accessor methods.

syntax:

```
def  getVariable(self):
    return self.variable
```

# 2. Class Methods:

Inside method implementation if we are using only class variables (static variables), then such type of methods we should declare as class method.

We can declare class method explicitly by using @classmethod decorator.
For class method we should provide cls variable at the time of declaration
We can call classmethod by using classname or object reference variable.

## 3. Static Methods:

In general these methods are general utility methods.
Inside these methods we won't use any instance or class variables.
Here we won't provide self or cls arguments at the time of declaration.

We can declare static method explicitly by using @staticmethod decorator
We can access static methods by using classname or object reference

Note: In general we can use only instance and static methods.Inside static method we can access class level variables by using class name.

class methods are most rarely used methods in python.

## Passing members of one class to another class:

We can access members of one class inside another class.

## Inner classes:

Sometimes we can declare a class inside another class,such type of classes are called inner classes.

Without existing one type of object if there is no chance of existing another type of object,then we should go for inner classes.

Note: Without existing outer class object there is no chance of existing inner class object. Hence inner class object is always associated with outer class object.

Note: The following are various possible syntaxes for calling inner class method

```
1.
o=Outer()
i=o.Inner()
i.m1()

2.
i=Outer().Inner()
i.m1()

3. Outer().Inner().m1()
```

## How to enable and disable Garbage Collector in our program:

By default Gargbage collector is enabled, but we can disable based on our requirement. In this context we can use the following functions of gc module.

1. gc.isenabled()
   Returns True if GC enabled

2. gc.disable()
   To disable GC explicitly

3. gc.enable()
   To enable GC explicitly