**Example for Class:**

class MyClass:

"This is my second class"

a = 10

def func(self):

print('Hello')

# Output: 10

print(MyClass.a)

# Output: <function MyClass.func at 0x0000000003079BF8>

print(MyClass.func)

# Output: 'This is my second class'

print(MyClass.\_\_doc\_\_)

**Example for Object:**

class MyClass:

"This is my second class"

a = 10

def func(self):

print('Hello')

# create a new MyClass

ob = MyClass()

# Output: <function MyClass.func at 0x000000000335B0D0>

print(MyClass.func)

# Output: <bound method MyClass.func of <\_\_main\_\_.MyClass object at 0x000000000332DEF0>>

print(ob.func)

# Calling function func()

# Output: Hello

ob.func()

class ComplexNumber:

def \_\_init\_\_(self,r = 0,i = 0):

self.real = r

self.imag = i

def getData(self):

print("{0}+{1}j".format(self.real,self.imag))

# Create a new ComplexNumber object

c1 = ComplexNumber(2,3)

# Call getData() function

# Output: 2+3j

c1.getData()

# Create another ComplexNumber object

# and create a new attribute 'attr'

c2 = ComplexNumber(5)

c2.attr = 10

# Output: (5, 0, 10)

print((c2.real, c2.imag, c2.attr))

# but c1 object doesn't have attribute 'attr'

# AttributeError: 'ComplexNumber' object has no attribute 'attr'

c1.attr

**Example for Inheritance:**

class Polygon:

def \_\_init\_\_(self, no\_of\_sides):

self.n = no\_of\_sides

self.sides = [0 for i in range(no\_of\_sides)]

def inputSides(self):

self.sides = [float(input("Enter side "+str(i+1)+" : ")) for i in range(self.n)]

def dispSides(self):

for i in range(self.n):

print("Side",i+1,"is",self.sides[i])

class Triangle(Polygon):

def \_\_init\_\_(self):

Polygon.\_\_init\_\_(self,3)

def findArea(self):

a, b, c = self.sides

# calculate the semi-perimeter

s = (a + b + c) / 2

area = (s\*(s-a)\*(s-b)\*(s-c)) \*\* 0.5

print('The area of the triangle is %0.2f' %area)

>>> t = Triangle()

>>> t.inputSides()

Enter side 1 : 3

Enter side 2 : 5

Enter side 3 : 4

>>> t.dispSides()

Side 1 is 3.0

Side 2 is 5.0

Side 3 is 4.0

>>> t.findArea()

The area of the triangle is 6.00

**Example for Operator Overloading:**

class Point:

def \_\_init\_\_(self, x = 0, y = 0):

self.x = x

self.y = y

def \_\_str\_\_(self):

return "({0},{1})".format(self.x,self.y)

def \_\_add\_\_(self,other):

x = self.x + other.x

y = self.y + other.y

return Point(x,y)

>>> p1 = Point(2,3)

>>> p2 = Point(-1,2)

>>> print(p1 + p2)

(1,5)