In [6]:

```
# 1) Create a Python class named Circle constructed by a radius and two
# methods which will compute the area and the perimeter of a circle.

class Circle():
 def __init__(self, r):
    self.radius = r
 def area(self):
    return self.radius ** 2 * 3.14
 def perimeter(self):
        return 2 * self.radius * 3.14



NewCircle = Circle(float(input("Please Enter Radius of Circle : ")))
print("Area of Circle is :{} ".format(NewCircle.area()))
print("Perimeter of Circle is :{} ".format(NewCircle.perimeter()))
```

```
Please Enter Radius of Circle : 5
Area of Circle is :78.5
Perimeter of Circle is :31.400000000000002
```

In [7]:

```
#2) Create a Python class named Rectangle constructed by a length and width
# and a method which will compute the area of a rectangle.

class Rectangle:
 def __init__(self, length, width):
    self.length = length
    self.width = width
 def area(self):
    return self.length * self.width

l=float(input("Please Enter Length of Rectangle : "))
w=float(input("Please Enter width of Rectangle : "))

r = Rectangle(l, w)
print("Area of Rectangle is :{} ".format(r.area()))
```

```
Please Enter Length of Rectangle : 4
Please Enter width of Rectangle : 5
Area of Rectangle is :20.0
```

In [31]:

```
# 3)Write a program to implement constructor.
import operator
from operator import eq, add, sub,mul

class calculator:
    first, second = 0, 0
    def __init__(self, f, s):
        self.first = f
        self.second = s
    def calculate(self):
        print("Answer is ",self.first+self.second)

n1=int(input("Enter No one : "))
n2=int(input("Enter No two : "))
obj = calculator(n1,n2)
obj.calculate()
```

```
Enter No one : 4
Enter No two : 5
Answer is  9
```

In [38]:

```
#4) Write a program to implement multiple inheritance.

class Class1:
    def m(self):
        print("In Class1")
class Class2(Class1):
    def m(self):
        print("In Class2")
class Class3(Class1):
    def m(self):
        print("In Class3")
class Class4(Class2, Class3):
    pass


Class1.m(1)
Class2.m(1)
Class3.m(1)
Class4.m(1)
```

```
In Class1
In Class2
In Class3
In Class2
```

In [40]:

```
# 5) Write a program to implement polymorphism.

def add(x, y, z=3):
    return x + y + z


print(add(6, 7))
print(add(2, 5, 6))
```

```
16
13
```

In [53]:

```
# 6) Write a program to implement method overriding.

class Fruits:
    def show(self):
        print("This is Fruits class")
class Apple(Fruits):
    def show(self):
        print("Apple")

a=Apple()
a.show()
f=Fruits()
f.show()
```

```
Apple
This is Fruits class
```

In [56]:

```
# 7) Write a program to implement composition
class A:
    def __init__(self):
        print('Class - A Contructor')
    def m1(self):
        print('M1 method of Class - A.')
class B:
    def __init__(self):
        print('Class - B Constructor.')
    def m2(self):
        obj = A()
        obj.m1()
        print('M2 method of Class - B.')


obj = B()
obj.m2()
```

```
Class - B Constructor.
Class - A Contructor
M1 method of Class - A.
M2 method of Class - B.
```

In [59]:

```
# 8) Write a program to implement aggregation
class Heart:
    def __init__(self, heartValves):
        self.heartValves = heartValves
    def display(self):
        return self.heartValves
class Person:
    def __init__(self, fname, heartValves):
        self.fname = fname
        self.heartValves = heartValves # Aggregation
    def display(self):
        print("First Name: ", self.fname)
        print("No of Healthy Valves: ", hv.display())


hv = Heart(4)
p = Person("ABC", hv)
p.display()
```

```
First Name:  ABC
No of Healthy Valves:  4
```

In [55]:

```
# 9) Write a program to implement abstraction.
class One:
    def disp(self):
        pass
class Two(One):
    def disp(self):
        print("Abstract method body in sub Class")

obj = Two()
obj.disp()
```

```
Abstract method body in sub Class
```

In [62]:

```
#10) Write a program to implement to show the use of access specifier.



class User:
    def __init__(self, name, age):
        self.usrName = name
        self.usrAge = age

    def displayAge(self):
        print("Age: ", self.usrAge)

obj = User("ABC", 20)

print("Name: ", obj.usrName)
obj.displayAge()
```

```
Name:  ABC
Age:  20
```

In [ ]: