In [4]:

```
# Example 01
class CSStudent:
    dept = 'Software Engineering'   # class variable

    def __init__(self, roll_num):
        self.roll_num = roll_num

student1 = CSStudent(80)
print(student1.dept)
print(student1.roll_num)
```

```
Software Engineering
80
```

In [10]:

```
# Example 02
class CSStudent:
    dept1 = 'Software Engineering'
    dept2 = 'Computer Science'

    def __init__(self, roll):
        self.roll = roll

    def setAddress(self, address):
        self.address = address

    def getAddress(self):
        return self.address

student2 = CSStudent(80)
print(student2.dept1)
student2.setAddress("Gulshan-e-Iqbal")
print(student2.getAddress())

student3 = CSStudent(90)
print(student3.dept2)
student3.setAddress('Saddar')
print(student3.getAddress())
```

```
Software Engineering
Gulshan-e-Iqbal
Computer Science
Saddar
```

In [12]:

```
# Example 03
# access modifiers (private attributes)
class MyClass:
    __hiddenVar = 0

    def add(self, increment):
        self.__hiddenVar += increment
        print (self.__hiddenVar)

ob1 = MyClass()
ob1.add(5)
ob1.add(5)
print(ob1.__hiddenVar)
```

```
5
10

---------------------------------------------------------------------------
-
AttributeError                            Traceback (most recent call las
t)
<ipython-input-12-1b9fa24e8bab> in <module>
     11 ob1.add(5)
     12 ob1.add(5)
---> 13 print(ob1.__hiddenVar)

AttributeError: 'MyClass' object has no attribute '__hiddenVar'
```

The above result gives error because we cannot access private attribute directly

In [15]:

```
# Example 04
class MyClass:
    __hiddenVar = 20

ob1 = MyClass()
print(ob1._MyClass__hiddenVar)  # Accessing Private attribute
```

```
20
```

In [19]:

```
# Example 05
class Person:
    def __init__(a, name, age, profession):
        a.name = name
        a.age = age
        a.profession = profession

    def mydetail(a):
        return ("Assalam o Alekum, my name is " + a.name +" and my age is " + str(a.age
) +
                ' and i am a ' + a.profession)

p1 = Person("Abdul Wassay", 18, "Student")
print(p1.mydetail())
```

```
Assalam o Alekum, my name is Abdul Wassay and my age is 18 and i am a Stud
ent
```

In [26]:

```
# Example 06
class Person:
    def __init__(a, name, age, profession):
        a.name = name
        a.age = age
        a.profession = profession

    def mydetail(a):
        return ("Assalam o Alekum, my name is " + a.name +", my age is " + str(a.age) +
                ' and i am ' + a.profession)

p2 = Person("Abdul Wassay", 18, "student")
print('Before modifying attributes:')
print(p2.mydetail())

print('\nAfter modifying attributes:')
p2.name = 'S.Faisal Ali'
p2.age = 55
p2.profession = 'Asst.Professor'
print(p2.mydetail())
```

```
Before modifying attributes:
Assalam o Alekum, my name is Abdul Wassay, my age is 18 and i am student

After modifying attributes:
Assalam o Alekum, my name is S.Faisal Ali, my age is 55 and i am Asst.Prof
essor
```

In [30]:

```
# Example 07
class Parent:
    parentAttr = 20
    def __init__(self):
        print ("Calling parent constructor")
    def parentMethod(self):
        print ('Calling parent method')
    def setAttr(self, attr):
        Parent.parentAttr = attr
    def getAttr(self):
        print ("Parent attribute :", Parent.parentAttr)

class Child(Parent):
    def __init__(self):
        print ("Calling child constructor")
    def childMethod(self):
        print ('Calling child method')

c1 = Child()
c1.childMethod()
c1.parentMethod()
c1.setAttr(200)
c1.getAttr()
```

```
Calling child constructor
Calling child method
Calling parent method
Parent attribute : 200
```

In [41]:

```
# Task 02
class Shape:
    def __init__(self, color=None):
        self.color = color
    def setColor(self, color):
        self.color = color
    def getColor(self):
        return self.color

class Circle(Shape):
    def __init__(self, radius=1):
        self.radius = radius
    def setRadius(self, radius):
        self.radius = radius
    def getRadius(self):
        return self.radius
    def getArea(self):
        return (3.14 * (self.radius**2))
    def getCircumference(self):
        return (2 * (3.14 * self.radius))

class Rectangle(Shape):
    def __init__(self, length=1, width=2):
        self.length = length
        self.width = width
    def setLength(self, length):
        self.length = length
    def getLength(self):
        return self.length
    def setWidth(self, width):
        self.width = width
    def getWidth(self):
        return self.width
    def getArea(self):
        return (self.length * self.width)
    def getPerimeter(self):
        return (2 * (self.length + self.width))
```

In [43]:

```
# Circle
c1 = Circle()
c1.setColor('Orange')
c1.setRadius(30)
print('Color:', c1.getColor())
print('Radius', c1.getRadius())
print('Area', c1.getArea())
print('Circumference:', c1.getCircumference())
```

```
Color: Orange
Radius 30
Area 2826.0
Circumference: 188.4
```

In [45]:

```
# Rectangle
r1 = Rectangle()
r1.setColor('Blue')
r1.setLength(20)
r1.setWidth(40)
print('Color:', r1.getColor())
print('Length:', r1.getLength())
print('Width:', r1.getWidth())
print('Area:', r1.getArea())
print('Perimeter:', r1.getPerimeter())
```

```
Color: Blue
Length: 20
Width: 40
Area: 800
Perimeter: 120
```

In [15]:

```
# Task 03
class Courses:
    def __init__(self):
        self.courses_list = []

    def enrollCourses(self, *courses):
        for course in courses:
            self.courses_list.append(course)

    def removeCourses(self, *courses):
        for course in courses:
            self.courses_list.remove(course)

    def getCourses(self):
        return self.courses_list

class CSProgram(Courses):
    def __init__(self, name, roll):
        self.name = name
        self.roll = roll
        Courses.__init__(self)

    def details(self):
        print(f'Name: {self.name}\nRoll No. {self.roll}\nEnrolled Courses: {self.course
s_list}')

class SEProgram(Courses):
    def __init__(self, name, roll):
        self.name = name
        self.roll = roll
        Courses.__init__(self)

    def details(self):
        print(f'Name: {self.name}\nRoll No. {self.roll}\nEnrolled Courses: {self.course
s_list}')
```

In [20]:

```
cs1 = CSProgram('Ali',36)
cs1.enrollCourses('English', 'Electronics', 'Programing', 'Calculus')
cs1.details()
```

```
Name: Ali
Roll No. 36
Enrolled Courses: ['English', 'Electronics', 'Programing', 'Calculus']
```

In [21]:

```
se1 = SEProgram('Abdul Wassay',80)
se1.enrollCourses('Electronics', 'Programing', 'Calculus')
se1.details()
```

```
Name: Abdul Wassay
Roll No. 80
Enrolled Courses: ['Electronics', 'Programing', 'Calculus']
```

In [41]:

```
# Task 04
class Bike:
    def __init__(self, name, engine, suspension, transmission, wheels, price):
        self.name = name
        self.engine = engine
        self.suspension = suspension
        self.transmission = transmission
        self.wheels = wheels
        self.price = price

class X75_Hurricane(Bike):
    def __init__(self,name,engine,suspension,transmission,wheels,price):
        Bike.__init__(self, name, engine, suspension, transmission, wheels, price)

    def details(self):
        print("The name of Bike is:", self.name)
        print("The engine of SuperHawk is:", self.engine)
        print("The suspension of SuperHawk is:", self.suspension)
        print("The transmission of SuperHawk is:", self.transmission)
        print("The wheels of SuperHawk is:", self.wheels)
        print("The price of SuperHawk is:", self.price)

class SuperHawk(Bike):
    def __init__(self,name,engine,suspension,transmission,wheels,price):
        Bike.__init__(self, name, engine, suspension, transmission, wheels, price)


    def details_SuperHawk(self):
        print("The name of Bike is:", self.name)
        print("The engine of SuperHawk is:", self.engine)
        print("The suspension of SuperHawk is:", self.suspension)
        print("The transmission of SuperHawk is:", self.transmission)
        print("The wheels of SuperHawk is:", self.wheels)
        print("The price of SuperHawk is:", self.price)

class Venom(Bike):
    def __init__(self,name,engine,suspension,transmission,wheels,price):
        Bike.__init__(self, name, engine, suspension, transmission, wheels, price)

    def details_Venom(self):
        print("The name of Bike is:", self.name)
        print("The engine of Venom is:", self.engine)
        print("The suspension of Venom is:", self.suspension)
        print("The transmission of Venom is:", self.transmission)
        print("The wheels of Venom is:", self.wheels)
        print("The price of Venom is:", self.price)
```

In [42]:

```
bike1 = X75_Hurricane("X 75 Hurricane"," USMC M1030 M1","fork tubes","Manual","Wire Whe
el","2 lac")
bike1.details()
```

```
The name of Bike is: X 75 Hurricane
The engine of SuperHawk is:  USMC M1030 M1
The suspension of SuperHawk is: fork tubes
The transmission of SuperHawk is: Manual
The wheels of SuperHawk is: Wire Wheel
The price of SuperHawk is: 2 lac
```

In [45]:

```
bike2 = SuperHawk("Super Hawk","Chevrolet V8 engine","Back tubes","AUTOMATIC","Wire Whe
el","5 lac")
bike2.details_SuperHawk()
```

```
The name of Bike is: Super Hawk
The engine of SuperHawk is: Chevrolet V8 engine
The suspension of SuperHawk is: Back tubes
The transmission of SuperHawk is: AUTOMATIC
The wheels of SuperHawk is: Wire Wheel
The price of SuperHawk is: 5 lac
```

In [46]:

```
bike3 = Venom("Venom","Wankel engine","Rear suspension","Manual through clutch","One pi
ece Wheel","1.5 lac")
bike3.details_Venom()
```

```
The name of Bike is: Venom
The engine of Venom is: Wankel engine
The suspension of Venom is: Rear suspension
The transmission of Venom is: Manual through clutch
The wheels of Venom is: One piece Wheel
The price of Venom is: 1.5 lac
```