Question 1

In [4]:
```
class Student:

    ID = 0

    def __init__(self,name,dept,age,cgpa):
        self.name = name
        self.dept = dept
        self.age = age
        self.cgpa = cgpa
        Student.ID += 1

    def get_details(self):
        print('ID',Student.ID)
        print('Name:',self.name)
        print('Department:',self.dept)
        print('Age:',self.age)
        print('CGPA:',self.cgpa)

    @classmethod
    def from_String(cls,info):
        name,dept,age,cgpa = info.split('-')
        obj = cls(name,dept,age,cgpa)
        return obj

s1 = Student("Samin", "CSE", 21, 3.91)
s1.get_details()
print("-----------------------")
s2 = Student("Fahim", "ECE", 21, 3.85)
s2.get_details()
print("-----------------------")
s3 = Student("Tahura", "EEE", 22, 3.01)
s3.get_details()
print("-----------------------")
s4 = Student.from_String("Sumaiya-BBA-23-3.96")
s4.get_details()
print("-----------------------")
print('Variables which are assigned values inside class methods are instance variables and All variables which a
print("-----------------------")
print('Instance methods can access the attributes and the other methods through the “self” parameter. Thus, inst
```

```
ID 1
```

```
Name: Samin
Department: CSE
Age: 21
CGPA: 3.91
-----------------------
ID 2
Name: Fahim
Department: ECE
Age: 21
CGPA: 3.85
-----------------------
ID 3
Name: Tahura
Department: EEE
Age: 22
CGPA: 3.01
-----------------------
ID 4
Name: Sumaiya
Department: BBA
Age: 23
CGPA: 3.96
-----------------------
Variables which are assigned values inside class methods are instance variables and All variables which are
assigned a value in the class declaration are class variables. Instance variables or non-static variable are
owned by instances of the class i.e for each object or instance of a class, the instance variables are diffe
rent where Class or Static variables are the variables that belong to the class and not to objects and these
variables are shared amongst objects of the class
-----------------------
Instance methods can access the attributes and the other methods through the “self” parameter. Thus, instanc
e methods can be only invoked through an instance of the class. On the other hand, The class method takes a
different parameter “cls” instead of “self” that points to the class instead of the object. Hence, it can’t
modify the state of an instance but can modify the state of a class.
```

Question 2

In [6]:
```
class Dates:

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

    @classmethod
    def toDashDate(cls,info):
        date = info.replace("/", "-")
        obj = cls(date)
        return obj


    def getDate(self):
        return self.date


date1 = Dates("05-09-2020")
dateFromDB = "05/09/2020"
date2= Dates.toDashDate(dateFromDB)
if(date1.getDate() == date2.getDate()):
    print("Equal")
else:
    print("Unequal")
print('=========================')
print("In the 1st object date1, between day,month and year there is hypen(-) and in the 2nd object date2, there
```

```
Equal
=========================
In the 1st object date1, between day,month and year there is hypen(-) and in the 2nd object date2, there is sl
ash(/). Now we replace the slash by hypen in the classmethod. And using getDate we return the value. When we c
ompare the two values, we get the same result and that's why we get equal as output
```

Question 3

In [5]:
```
class Passenger:

    count = 0

    def __init__(self,name):
        self.name = name
        Passenger.count += 1

    def set_bag_weight(self,weight):
        self.weight = weight

    def printDetail(self):
        print('Name:',self.name)

        if self.weight <= 20:
            fare = 450
            print('Bus Fare:',fare,'taka')

        elif self.weight >= 21 and self.weight <=50:
            fare = 450+50
            print('Bus Fare:',fare,'taka')

        else:
            fare = 450+100
            print('Bus Fare:',fare,'taka')

print('Total Passenger:', Passenger.count)
p1 = Passenger('Jack')
p1.set_bag_weight(90)
p2 = Passenger('Carol')
p2.set_bag_weight(10)
p3 = Passenger('Mike')
p3.set_bag_weight(25)
print("========================")
p1.printDetail()
print("========================")
p2.printDetail()
print("========================")
p3.printDetail()
print("========================")
print('Total Passenger:', Passenger.count)
```

```
Total Passenger: 0
=========================
Name: Jack
Bus Fare: 550 taka
=========================
Name: Carol
Bus Fare: 450 taka
=========================
Name: Mike
Bus Fare: 500 taka
=========================
Total Passenger: 3
```

Question 4

In [6]:
```
class Travel:

    count = 0

    def __init__(self,source,destination,time=1):
        self.source = source
        self.destination = destination
        self.time = time
        Travel.count += 1

    def set_source(self,source):
        self.source = source

    def set_destination(self,destination):
        self.destination = destination

    def set_time(self,time):
        self.time = time

    def display_travel_info(self):
        return f"Source:{self.source}\nDestination:{self.destination}\nFligth time:{self.time}:00\n"


print('No. of Traveller =', Travel.count)
print("======================")
t1 = Travel("Dhaka","India")
print(t1.display_travel_info())
print("======================")
t2 = Travel("Kuala Lampur","Dhaka")
t2.set_time(23)
print(t2.display_travel_info())
print("======================")
t3 = Travel("Dhaka","New_Zealand")
t3.set_time(15)
t3.set_destination("Germany")
print(t3.display_travel_info())
print("======================")
t4 = Travel("Dhaka","India")
t4.set_time(9)
t4.set_source("Malaysia")
t4.set_destination("Canada")
print(t4.display_travel_info())
```

```
print("======================")
print('No. of Traveller =', Travel.count)
```

```
No. of Traveller = 0
======================
Source:Dhaka
Destination:India
Fligth time:1:00

======================
Source:Kuala Lampur
Destination:Dhaka
Fligth time:23:00

======================
Source:Dhaka
Destination:Germany
Fligth time:15:00

======================
Source:Malaysia
Destination:Canada
Fligth time:9:00

======================
No. of Traveller = 4
```

Question 5

In [7]:
```
class Fruit:

    fruitCount = 0

    def __init__(self,name,count):
        self.name = name
        self.count = count
        Fruit.fruitCount = Fruit.fruitCount + self.count

    @classmethod
    def saySomethingGood(self):
        print('Fruits are good for health')

    @classmethod
    def resetcount(self):
        Fruit.fruitCount = 0

apples = Fruit("Apple", 3);
pears = Fruit("Pear", 4);
print(apples.name, apples.count)
print(pears.name,pears.count)
print("Total number of fruits", Fruit.fruitCount)
Fruit.saySomethingGood()
Fruit.resetcount()
print("Total number of fruits", Fruit.fruitCount)
```

```
Apple 3
Pear 4
Total number of fruits 7
Fruits are good for health
Total number of fruits 0
```

Question 6

In [2]:
```
class Cat:

    Number_of_cats = 0

    def __init__(self,color,work):
        self.color = color
        self.work = work
        Cat.Number_of_cats += 1

    @classmethod
    def no_parameter(cls,color='White',work='sitting'):
        cls.color = color
        cls.work = work
        obj = cls(color,work)
        return obj
        Cat.Number_of_cats += 1

    @classmethod
    def first_parameter(cls,color='Black',work='sitting'):
        cls.color = color
        cls.work = work
        obj = cls(color,work)
        return obj
        Cat.Number_of_cats += 1

    @classmethod
    def second_parameter(cls,work='sitting',color='Grey',):
        cls.color = color
        cls.work = work
        obj = cls(color,work)
        return obj
        Cat.Number_of_cats += 1

    def changeColor(self,color):
        self.color = color

    def printCat(self):
        print(self.color,'cat is',self.work)

print("Total number of cats:", Cat.Number_of_cats)
c1 = Cat.no_parameter()
c2 = Cat.first_parameter("Black")
```

```
c3 = Cat("Brown", "jumping")
c4 = Cat("Red", "purring")
c5 = Cat.second_parameter("playing")
print("======================")
c1.printCat()
c2.printCat()
c3.printCat()
c4.printCat()
c5.printCat()
c1.changeColor("Blue")
c3.changeColor("Purple")
c1.printCat()
c3.printCat()
print("======================")
print("Total number of cats:", Cat.Number_of_cats)
```

```
Total number of cats: 0
======================
White cat is sitting
Black cat is sitting
Brown cat is jumping
Red cat is purring
Grey cat is playing
Blue cat is sitting
Purple cat is jumping
======================
Total number of cats: 5
```

Question 7

In [7]:
```
from math import pi
class Cylinder:

    radius = 5
    height = 18

    def __init__(self,radius,height):

        self.radius = radius
        self.height = height
        print('Default radius= '+str(Cylinder.radius)+' and height= '+str(Cylinder.height)+'.')
        print('Updated radius= '+str(self.radius)+' and height= '+str(self.height)+'.')
        Cylinder.radius = self.radius
        Cylinder.height = self.height

    @staticmethod
    def area(radius,height):
        area = 2*pi*radius*radius + 2*pi*radius*height
        print('Area:',area)

    @staticmethod
    def volume(radius,height):
        volume = pi*radius*radius*height
        print('Volume:',volume)

    @classmethod
    def swap(cls,height,radius):
        obj = cls(radius,height)
        return obj

    @classmethod
    def changeFormat(cls,info):
        radius,height = info.split('-')
        obj = cls(float(radius),float(height))
        return obj

c1 = Cylinder(0,0)
Cylinder.area(c1.radius,c1.height)
Cylinder.volume(c1.radius,c1.height)
print("==============================")
c2 = Cylinder.swap(8,3)
c2.area(c2.radius,c2.height)
```

```
c2.volume(c2.radius,c2.height)
print("==============================")
c3 = Cylinder.changeFormat("7-13")
c3.area(c3.radius,c3.height)
c3.volume(c3.radius,c3.height)
print("==============================")
Cylinder(0.3,5.56).area(Cylinder.radius,Cylinder.height)
print("==============================")
Cylinder(3,5).volume(Cylinder.radius,Cylinder.height)
```

```
Default radius= 5 and height= 18.
Updated radius= 0 and height= 0.
Area: 0.0
Volume: 0.0
==============================
Default radius= 0 and height= 0.
Updated radius= 3 and height= 8.
Area: 207.34511513692635
Volume: 226.1946710584651
==============================
Default radius= 3 and height= 8.
Updated radius= 7.0 and height= 13.0.
Area: 879.645943005142
Volume: 2001.1945203366981
==============================
Default radius= 7.0 and height= 13.0.
Updated radius= 0.3 and height= 5.56.
Area: 11.045839770021713
==============================
Default radius= 0.3 and height= 5.56.
Updated radius= 3 and height= 5.
Volume: 141.3716694115407
```

In [ ]: