In [2]:
```
###1 1. Chef is a software developer, so he has to switch between different languages sometime
###  has some features, which are represented by integers here.
###  Currently, Chef has to use a language with two given features A and B. He has two options
###  with two features A1 and B1, or to a language with two features A2 and B2. All four featu
###  are pairwise distinct.
###  Tell Chef whether he can use the first language, the second language or neither of these

###  The first and only line of each test case contains six space-separated integers A,B,A1,B1

###  For each test case, print a single line containing the integer 1 if Chef should switch to
###  or 2 if Chef should switch to the second language, or 0 if Chef cannot switch to either l

a,b,a1,b1,a2,b2 = map(int,input("Enter a,b,a1,b1,a2,b2:").split())
if((a==a1 and b==b1)or(a==b1 and b==a1)):
  print("1")
elif((a==a2 and b==b2)or(a==b2 and b==a2)):
  print("2")
else:
  print("0")
```

```
Enter a,b,a1,b1,a2,b2:2 2 1 2 1 2
0
```

In [3]:
```
###2. You have prepared four problems. The difficulty levels of the problems are A1,A2,A3,A4 r
###   A problem set comprises two problems and no two problems in a problem set should have th
###   A problem can belong to at most one problem set. Find the maximum number of problem sets
###   four problems.
###   Each test case contains four space-separated integers A1, A2, A3, A4, denoting the diffi
###   For each test case, print a single line containing one integer - the maximum number of p
###   you can create using the four problems.

a1,a2,a3,a4 = map(int,input("Enter a1,a2,a3,a4:").split())
if(a1==a2==a3==a4):
  print("0")
elif(a1==a2==a3 or a1==a2==a4 or a1==a3==a4 or a2==a3==a4):
  print("1")
else:
  print("2")
```

```
Enter a1,a2,a3,a4:2 2 1 2
1
```

In [4]:
```
###3. Develop a python code to check given two dates d1 and d1 , check whether d1 is less than
###   is greater than d2 or d1 is equal to d2. (Hint: overload < , > , == operators)

import datetime

### date in yyyy/mm/dd format
d1 = datetime.datetime(2022, 9, 30)
d2 = datetime.datetime(2022, 11, 20)

### Comparing the dates will return
### Check True or False
print("d1 is less than d2 : ", d1 < d2)
print("d1 is greater than d2 : ", d1 > d2)
print("d1 is not equal to d2 : ", d1 != d2)
```

```
d1 is less than d2 :  True
d1 is greater than d2 :  False
d1 is not equal to d2 :  True
```

In [9]:
```
###   Develop python code to add, subtract , multiply and divide two distances where each dist
###   two things of the format KM followed by Meters. (Example: d1 = 4km 500 m  and d2 = 3 km

class Distance:
    def GetDistance(self):
        self.__km = int(input("Enter KM: "))
        self.__m=int(input("Enter M: "))

    def PutDistance(self):
```

```
        print(self.__km,self.__m)

    def __add__(self, T):
        R=Distance()
        R.__m=self.__m+T.__m
        R.__km = self.__km + T.__km
        R.__m=R.__m
        R.__km=R.__km+(R.__m//1000)
        R.__m=R.__m%1000
        return R

    def __sub__(self, T):
        R=Distance()
        R.__m=self.__m-T.__m
        R.__km = self.__km-T.__km
        R.__m=R.__m
        R.__km=R.__km-(R.__m//1000)
        R.__m=R.__m%1000
        return R

    def __mul__(self, T):
        R=Distance()
        R.__m=self.__m*T.__m
        R.__km = self.__km*T.__km
        R.__m=R.__m
        R.__km=R.__km
        return R

    def __div__(self, T):
        R=Distance()
        R.__m=self.__m//T.__m
        R.__km = self.__km//T.__km
        R.__m=R.__m
        R.__km=R.__km
        return R

D1=Distance()
D2=Distance()
print("Enter first distance")
D1.GetDistance()
print("Enter second distance")
D2.GetDistance()

D3=D1+D2
print("The sum of both distance is")
D3.PutDistance()

D4=D1-D2
print("The subtracted distance between D1-D2 is")
D4.PutDistance()

D5=D1*D2
print("The multiplied distance between D1*D2 is")
D5.PutDistance()

##"D6=D1/D2"
##"print(The division distance between D1 by D2 is)"
##D6.PutDistance()
```

```
Enter first distance
Enter KM: 4
Enter M: 400
Enter second distance
Enter KM: 2
Enter M: 200
The sum of both distance is
6 600
The subtracted distance between D1-D2 is
2 200
The multiplied distance between D1*D2 is
8 80000
```

In [10]:
```
###4 Develop a class called Box with attributes length, breadth, depth and define required cor
###  other relevant methods. Inherit Box class to WeightBox which contains extra attribute as
###  From this extent further as ColorWeightBox which has Color as extra attribute. Develop co
###  using multi-level inheritance.
class Box:
  def __init__(self,l,b,d):
    self.l=l
    self.b=b
    self.d=d
  def volume(self):
    return(self.l*self.b*self.d)
  def display(self):
    print("Details of the Box Object are:")
    print("Length: ",self.l)
    print("Breadth: ",self.b)
    print("Depth: ",self.d)
class ColoredBox(Box):
  def __init__(self,l,b,d,c):
    Box.__init__(self,l,b,d)
    self.c=c
  def display(self): # The display() method in the Box class is overridding here
    print("Details of the ColoredBox Object are:")
    print("Length: ",self.l)
    print("Breadth: ",self.b)
    print("Depth: ",self.d)
    print("Color: ",self.c)

cb = ColoredBox(20,30,40,'Green')
print(cb.volume())
cb.display()#display() in ColoredBox class will be invoked
```

```
24000
Details of the ColoredBox Object are:
Length:  20
Breadth:  30
Depth:  40
Color:  Green
```

In [11]:
```
### #Inheritance- Multiple Inheritance
class A:
    def __init__(self,a):
        self.a=a
    def display(self):
        print("a: ",self.a)
class B:
    def __init__(self,b):
        self.b=b
    def display(self):
        print("b: ",self.b)
class C(A,B):# C is inheriting Class A and Class B
    def __init__(self,a,b,c):
        A.__init__(self,a)
        B.__init__(self,b)
        self.c=c
    def display(self):
        A.display(self)
        B.display(self)
        print("c: ",self.c)
a1 = A(4)
a1.display()
b1 = B(9)
b1.display()
c1 = C(1,2,3)
c1.display()
```

```
a:  4
b:  9
a:  1
b:  2
c:  3
```

In [ ]: