Python: without numpy or sklearn

Q1: Given two matrices please print the product of those two matrices

```
Ex 1: A   = [[1 3 4]
             [2 5 7]
             [5 9 6]]
      B   = [[1 0 0]
             [0 1 0]
             [0 0 1]]
      A*B = [[1 3 4]
             [2 5 7]
             [5 9 6]]


Ex 2: A   = [[1 2]
             [3 4]]
      B   = [[1 2 3 4 5]
             [5 6 7 8 9]]
      A*B = [[11 14 17 20 23]
             [18 24 30 36 42]]


Ex 3: A   = [[1 2]
             [3 4]]
      B   = [[1 4]
             [5 6]
             [7 8]
             [9 6]]
      A*B =Not possible
```

In [1]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input examples
# you can free to change all these codes/structure
# here A and B are list of lists

def matrix_mul(A, B):
    """ This function multiplies two metrics and returns the result.
    """
    if (len(A[0]) != len(B)):
        return "Multiplication is Not possible"
    else:
        mult=[]
        #Loop for each row
        for i in range(len(A)):
            mtrow1=[]
            #Loop for each column
            for j in range(len(B[i])):
                mt=0
                #Loop to output individual elements of multiplication metrics
                for k in range(len(B)):
                    mt = mt + A[i][k]*B[k][j]
                mtrow1.append(mt)
            mult.append(mtrow1)
    return mult

# First Set of Matrics
A = [[1,3,4],[2,5,7],[5,9,6]]
B = [[1, 0, 0],[0, 1, 0],[0, 0, 1]]

# Second Set of Matrics
C = [[1, 2],[3, 4]]
D = [[1, 2, 3, 4, 5],[5, 6, 7, 8, 9]]

## Third Set of Matrics
E = [[1, 2],[3, 4]]
F = [[1, 4],[5, 6],[7, 8],[9, 6]]

#Print multiplied metrics
print("Multiplication of A*B is :", matrix_mul(A, B))
print("\nMultiplication of C*D is :", matrix_mul(C, D))
print("\nMultiplication of E*F is :",matrix_mul(E, F))
```

```
Multiplication of A*B is : [[1, 3, 4], [2, 5, 7], [5, 9, 6]]

Multiplication of C*D is : [[11, 14, 17, 20, 23], [23, 30, 37, 44, 51]]

Multiplication of E*F is : Multiplication is Not possible
```

Q2: Select a number randomly with probability proportional to its magnitude from the given array of n elements

consider an experiment, selecting an element from the list A randomly with probability proportional to its magnitude. assume we are doing the same experiment for 100 times with replacement, in each experiment you will print a number that is selected randomly from A.

```
Ex 1: A = [0 5 27 6 13 28 100 45 10 79]
let f(x) denote the number of times x getting selected in 100 experiments.
f(100) > f(79) > f(45) > f(28) > f(27) > f(13) > f(10) > f(6) > f(5) > f(0)
```

In [2]:
```
import random

# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input examples
# you can free to change all these codes/structure

def pick_a_number_from_list(A):
    num=random.sample(A,1)
    return num

def sampling_based_on_magnitude():
    sam=[]
    for i in range(1,100):
        number = pick_a_number_from_list(A)
        print(number)

# Given elements List
A = [0, 5, 27, 6, 13, 28, 100, 45, 10, 79]

# Extending list in proportion to weights of individual elements
for i in range(0,len(A)):
    A.extend([A[i]]*(A[i]-1))

# Calling sampling method to print sampled values as per their weights
sampling_based_on_magnitude()
```

```
[79]
[28]
[100]
[100]
[100]
[27]
[79]
[100]
[100]
[13]
[79]
[100]
[100]
[28]
```

```
[79]
[79]
[100]
[79]
[100]
[5]
[79]
[45]
[13]
[79]
[79]
[100]
[100]
[28]
[28]
[100]
[45]
[45]
[28]
[100]
[100]
[28]
[45]
[100]
[100]
[100]
[79]
[45]
[79]
[13]
[79]
[28]
[79]
[27]
[10]
[79]
[79]
[27]
[45]
[100]
[100]
[45]
[100]
```

```
[79]
[45]
[45]
[100]
[45]
[6]
[45]
[27]
[100]
[100]
[45]
[79]
[100]
[100]
[27]
[45]
[79]
[45]
[13]
[27]
[79]
[100]
[79]
[79]
[10]
[100]
[79]
[45]
[45]
[100]
[79]
[79]
[28]
[13]
[28]
[13]
[10]
[79]
[45]
[100]
[100]
[45]
```

Q3: Replace the digits in the string with

Consider a string that will have digits in that, we need to remove all the characters which are not digits and replace the digits with #

```
Ex 1: A = 234               Output: ###
Ex 2: A = a2b3c4            Output: ###
Ex 3: A = abc               Output:   (empty string)
Ex 5: A = #2a$#b%c%561#     Output: ####
```

In [3]:
```
import re
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input examples
# you can free to change all these codes/structure
# String: it will be the input to your program

def replace_digits(String):
    newsub=re.sub('[0-9]','#',re.sub('[a-z]','',String))
    #
    return newsub

print(replace_digits("234"))
print(replace_digits("a2b3c4"))
print(replace_digits("abc"))
print(replace_digits("#2a$#b%c%561#"))
```

```
###
###

##$#%%####
```

Q4: Students marks dashboard

Consider the marks list of class students given in two lists
Students = ['student1','student2','student3','student4','student5','student6','student7','student8','student9','student10']
Marks = [45, 78, 12, 14, 48, 43, 45, 98, 35, 80]
from the above two lists the Student[0] got Marks[0], Student[1] got Marks[1] and so on.

Your task is to print the name of students

a. Who got top 5 ranks, in the descending order of marks
b. Who got least 5 ranks, in the increasing order of marks
d. Who got marks between >25th percentile <75th percentile, in the increasing order of marks.

```
Ex 1:
Students=['student1','student2','student3','student4','student5','student6','student7','student8','stud
ent9','student10']
Marks = [45, 78, 12, 14, 48, 43, 47, 98, 35, 80]

a.
student8  98
student10 80
student2  78
student5  48
student7  47

b.
student3 12
student4 14
student9 35
student6 43
student1 45

c.
student9 35
student6 43
student1 45
student7 47
student5 48
```

In [4]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input examples
# you can free to change all these codes/structure

import pandas as pd

def display_dash_board(students, marks):
    """This function take students and corresponding marks in individual lists and returns the top top 5 student
    5 students and students between 25-75 percentile values
    """

    df=pd.DataFrame(data={'student':students,'Marks':marks},index=None)
    df_asc=df.sort_values(by=['Marks'],ascending=True)
    df_desc=df.sort_values(by=['Marks'],ascending=False)

    # write code for computing top top 5 students
    top_5_students = df_desc.head(5)

    # write code for computing top least 5 students
    least_5_students = df_asc.head(5)

    # write code for students within 25 to 75 percentile
    students_within_25_and_75 = df_asc.iloc[[i for i in range(int((.25*len(df))),int((.75*len(df))))]]

    #return dashboard values
    return top_5_students, least_5_students, students_within_25_and_75

Students = ['student1','student2','student3','student4','student5','student6','student7','student8','student9','
Marks = [45, 78, 12, 14, 48, 43, 45, 98, 35, 80]

top_5_students, least_5_students, students_within_25_and_75 = display_dash_board(Students, Marks)

print("Top Five Students are :\n\n",top_5_students.to_string(index=False,header=False),"\n\n Top Least 5 Student
least_5_students.to_string(index=False,header=False), "\n\n Students in 25-75 percentile range are : \n\n",stude
```

```
Top Five Students are :

   student8  98
  student10  80
```

```
 student2  78
 student5  48
 student1  45

Top Least 5 Students are :

 student3  12
student4  14
student9  35
student6  43
student1  45

Students in 25-75 percentile range are :

 student9  35
student6  43
student1  45
student7  45
student5  48
```

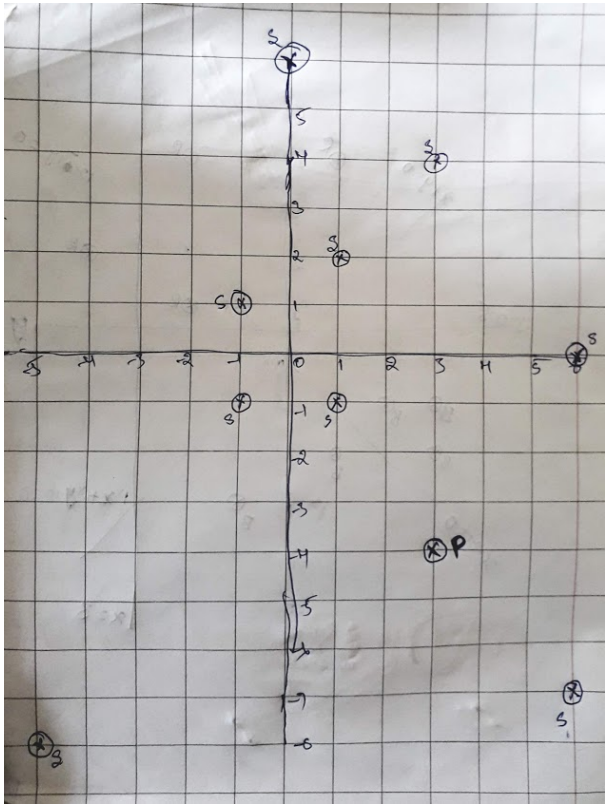
Q5: Find the closest points

Consider you are given n data points in the form of list of tuples like S=[(x1,y1),(x2,y2),(x3,y3),(x4,y4),(x5,y5),..,(xn,yn)] and a point P=(p,q)
your task is to find 5 closest points(based on cosine distance) in S from P

Cosine distance between two points (x,y) and (p,q) is defined as $cos^{-1}(\frac{(x\cdot p+y\cdot q)}{\sqrt{(x^2+y^2)}\cdot\sqrt{(p^2+q^2)}})$

Ex:

S= [(1,2),(3,4),(-1,1),(6,-7),(0, 6),(-5,-8),(-1,-1)(6,0),(1,-1)]
P= (3,-4)



Output:
(6,-7)
(1,-1)
(6,0)
(-5,-8)
(-1,-1)

In [5]:
```
import math

# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input examples
# you can free to change all these codes/structure
# here S is list of tuples and P is a tuple of len=2

def closest_points_to_p(S, P):
    """ This function returns 5 nearest points in S from point P
    """

    dict = {}
    for i in range(len(S)):
        x, y, p, q = S[i][0], S[i][1], P[0], P[1]

        #calculate distances
        dist = math.acos((x*p+y*q)/((math.sqrt(x**2+y**2))*(math.sqrt(p**2+q**2))))

        #Add to dictionary
        dict[(x,y)] = dist

        #sort based on distances
        dict3 = sorted((value, key) for (key,value) in dict.items())

    #return only 5 nearest points
    return [dict3[i][1] for i in range(5)]

S = [(1,2),(3,4),(-1,1),(6,-7),(0, 6),(-5,-8),(-1,-1),(6,0),(1,-1)]
P = (3,-4)

#Print closest points by Calling the function
print(closest_points_to_p(S, P))
```

```
[(6, -7), (1, -1), (6, 0), (-5, -8), (-1, -1)]
```

Q6: Find which line separates oranges and apples

Consider you are given two set of data points in the form of list of tuples like

```
Red =[(R11,R12),(R21,R22),(R31,R32),(R41,R42),(R51,R52),..,(Rn1,Rn2)]
Blue=[(B11,B12),(B21,B22),(B31,B32),(B41,B42),(B51,B52),..,(Bm1,Bm2)]
```

and set of line equations(in the string format, i.e list of strings)

```
Lines = [a1x+b1y+c1,a2x+b2y+c2,a3x+b3y+c3,a4x+b4y+c4,..,K lines]
Note: You need to do string parsing here and get the coefficients of x,y and intercept.
```

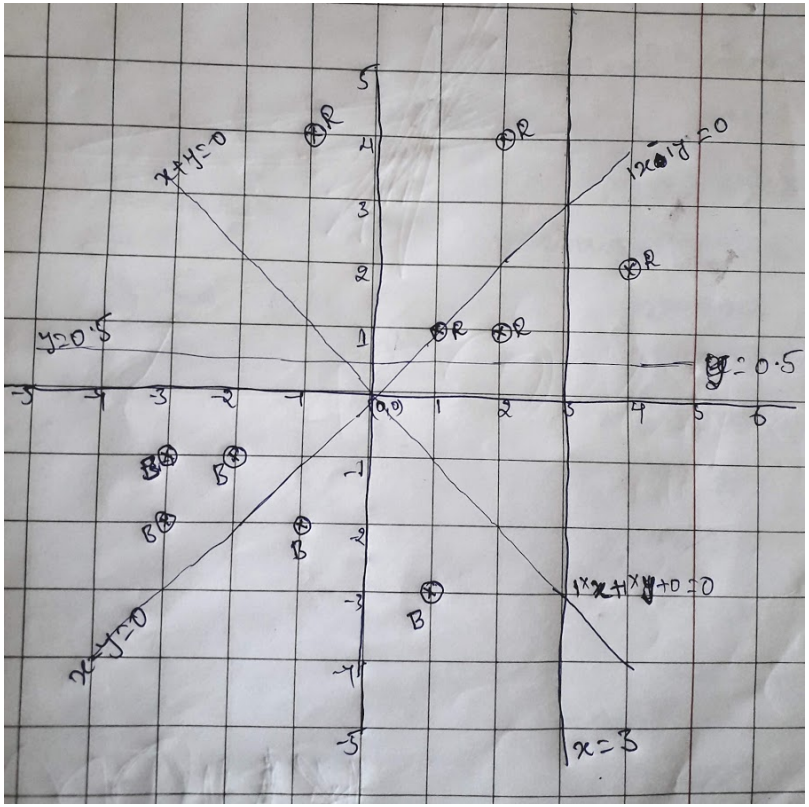
Your task here is to print "YES"/"NO" for each line given. You should print YES, if all the red points are one side of the line and blue points are on other side of the line, otherwise you should print NO.

Ex:

Red= [(1,1),(2,1),(4,2),(2,4), (-1,4)]

Blue= [(-2,-1),(-1,-2),(-3,-2),(-3,-1),(1,-3)]

Lines=["1x+1y+0","1x-1y+0","1x+0y-3","0x+1y-0.5"]



Output:

YES

NO

NO

YES

In [6]:
```
import math
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input strings
# you can free to change all these codes/structure

def i_am_the_one(red,blue,line):
    """This function finds whether the given set of blue and red points are correctly separated by the line
    """
    #Delcare variables
    rdotp, bdotp, rval, bval = [], [], [], []
    rflag, bflag = "No", "No"

    #Find w1 andf w2 values from line equations
    ws = line.split("x")
    w1 = int(ws[0])
    w2 = int(ws[1].split("y")[0])

    #Find the dot product of red point and line weigths vector
    for i in range(len(red)):
        rdotp.append(w1*(red[i][0]) + w2*(red[i][1]))
        rval.append("+ve" if rdotp[i] > 0 else "-ve")

    #Find the dot product of blue point and line weigths vector
    for j in range(len(blue)):
        bdotp.append(w1*(blue[j][0]) + w2*(blue[j][1]))
        bval.append("+ve" if bdotp[j] > 0 else "-ve")

    #Check if all red points are on same side
    for i in range(len(rval)-1):
        if rval[i] == rval [i+1]:
            rflag = 'Yes'

    #Check if all blue points are on same side
    for j in range(len(bval)-1):
        if bval[i] == bval[i+1]:
            bflag = 'Yes'

    # Finds if the red and blue points are separetd by line and returns 'Yes' or 'No' accordingly
    return 'Yes' if (rflag == 'Yes' and bflag == 'Yes' and rval[0] != bval[0]) else 'No'

Red= [(1,1),(2,1),(4,2),(2,4), (-1,4)]
```

```
Blue= [(-2,-1),(-1,-2),(-3,-2),(-3,-1),(1,-3)]
Lines=["1x+1y+0","1x-1y+0","1x+0y-3","0x+1y-0.5"]

for i in Lines:
    yes_or_no = i_am_the_one(Red, Blue, i)
    print(yes_or_no)
```

```
Yes
No
No
Yes
```

Q7: Filling the missing values in the specified format

You will be given a string with digits and '_'(missing value) symbols you have to replace the '_' symbols as explained

```
Ex 1: _, _, _, 24 ==> 24/4, 24/4, 24/4, 24/4 i.e we. have distributed the 24 equally to all 4 places

Ex 2: 40, _, _, _, 60 ==> (60+40)/5,(60+40)/5,(60+40)/5,(60+40)/5,(60+40)/5 ==> 20, 20, 20, 20, 20 i.e.
the sum of (60+40) is distributed qually to all 5 places

Ex 3: 80, _, _, _, _  ==> 80/5,80/5,80/5,80/5,80/5 ==> 16, 16, 16, 16, 16 i.e. the 80 is distributed qu
ally to all 5 missing values that are right to it

Ex 4: _, _, 30, _, _, _, 50, _, _
==> we will fill the missing values from left to right
    a. first we will distribute the 30 to left two missing values (10, 10, 10, _, _, _, 50, _, _)
    b. now distribute the sum (10+50) missing values in between (10, 10, 12, 12, 12, 12, 12, _, _)
    c. now we will distribute 12 to right side missing values (10, 10, 12, 12, 12, 12, 4, 4, 4)
```

for a given string with comma seprate values, which will have both missing values numbers like ex: "_, _, x, _, _, _" you need fill the missing values

Q: your program reads a string like ex: "_, _, x, _, _, _" and returns the filled sequence

Ex:

```
Input1: "_,_,_,24"
Output1: 6,6,6,6

Input2: "40,_,_,_,60"
Output2: 20,20,20,20,20

Input3: "80,_,_,_,_"
Output3: 16,16,16,16,16

Input4: "_,_,30,_,_,_,50,_,_"
Output4: 10,10,12,12,12,12,4,4,4
```

In [7]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input strings
# you can free to change all these codes/structure

def curve_smoothing(str):
    """This function fills missing values in a given digit string  by taking average of boundary values present
    """

    strsmooth=str.split(',')
    end = 0

    #while loop to check presence of underscoe '_'
    while(strsmooth.count('_') != 0):
        ind = strsmooth.index('_')

        #Fill missing values if start is missing
        if ind == 0:
            end = [ x for x in (range(len(strsmooth))) if strsmooth[x] != '_'][0]
            newval = int(int(strsmooth[end])/(end+1))
            for i in range(end+1):
                strsmooth[i]=newval

        #Fill missing values if end is missing
        elif ((ind + strsmooth.count('_')) == len(strsmooth)):
            blank = strsmooth.count('_')
            newval = int(int(strsmooth[ind-1])/(blank+1))
            for i in range((ind-1), (ind+blank)):
                strsmooth[i] = newval

        #Fill missing value in any other place
        elif ind != 0:
            for i in range(ind+1, len(strsmooth)):
                if strsmooth[i] != '_':
                    end=i
                    break
            newval = int((int(strsmooth[ind-1]) + int(strsmooth[end]))/(end-ind+2))
            for i in range(ind-1, end+1):
                strsmooth[i] = newval
    return strsmooth
```

```
S1 = "_,_,_,24"
S2 = "40,_,_,_,60"
S3 = "80,_,_,_,_"
S4 = "_,_,30,_,_,_,50,_,_"

print(" Input: \"{}\"".format(S1),"\n","Output:",curve_smoothing(S1),"\n")
print(" Input: \"{}\"".format(S2),"\n","Output:",curve_smoothing(S2),"\n")
print(" Input: \"{}\"".format(S3),"\n","Output:",curve_smoothing(S3),"\n")
print(" Input: \"{}\"".format(S4),"\n","Output:",curve_smoothing(S4),"\n")
```

```
 Input: "_,_,_,24"
 Output: [6, 6, 6, 6]

 Input: "40,_,_,_,60"
 Output: [20, 20, 20, 20, 20]

 Input: "80,_,_,_,_"
 Output: [16, 16, 16, 16, 16]

 Input: "_,_,30,_,_,_,50,_,_"
 Output: [10, 10, 12, 12, 12, 12, 4, 4, 4]
```

Q8: Find the probabilities

You will be given a list of lists, each sublist will be of length 2 i.e. [[x,y],[p,q],[l,m]..[r,s]] consider its like a martrix of n rows and two columns

1. The first column F will contain only 5 uniques values (F1, F2, F3, F4, F5)
2. The second column S will contain only 3 uniques values (S1, S2, S3)

```
your task is to find
a. Probability of P(F=F1|S==S1), P(F=F1|S==S2), P(F=F1|S==S3)
b. Probability of P(F=F2|S==S1), P(F=F2|S==S2), P(F=F2|S==S3)
c. Probability of P(F=F3|S==S1), P(F=F3|S==S2), P(F=F3|S==S3)
d. Probability of P(F=F4|S==S1), P(F=F4|S==S2), P(F=F4|S==S3)
e. Probability of P(F=F5|S==S1), P(F=F5|S==S2), P(F=F5|S==S3)
```

Ex:

```
[[F1,S1],[F2,S2],[F3,S3],[F1,S2],[F2,S3],[F3,S2],[F2,S1],[F4,S1],[F4,S3],[F5,S1]]
```

a. P(F=F1|S==S1)=1/4, P(F=F1|S==S2)=1/3, P(F=F1|S==S3)=0/3
b. P(F=F2|S==S1)=1/4, P(F=F2|S==S2)=1/3, P(F=F2|S==S3)=1/3
c. P(F=F3|S==S1)=0/4, P(F=F3|S==S2)=1/3, P(F=F3|S==S3)=1/3
d. P(F=F4|S==S1)=1/4, P(F=F4|S==S2)=0/3, P(F=F4|S==S3)=1/3
e. P(F=F5|S==S1)=1/4, P(F=F5|S==S2)=0/3, P(F=F5|S==S3)=0/3

In [8]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input strings
# you can free to change all these codes/structure

def compute_conditional_probabilites(A):
    """ This function calculates the probabilities of F values for any given S value
    """

    s1, s2, s3 = 0, 0, 0
    f1s1, f1s2, f1s3, f1s4, f1s5 = 0, 0, 0, 0, 0
    f2s1, f2s2, f2s3, f2s4, f2s5 = 0, 0, 0, 0, 0
    f3s1, f3s2, f3s3, f3s4, f3s5 = 0, 0, 0, 0, 0
    f4s1, f4s2, f4s3, f4s4, f4s5 = 0, 0, 0, 0, 0
    f5s1, f5s2, f5s3, f5s4, f5s5 = 0, 0, 0, 0, 0

    for item in A:
        if item[1] == 'S1':
            s1 = s1 + 1
            if item[0] == 'F1':
                f1s1 += 1
            if item[0] == 'F2':
                f2s1 += 1
            if item[0] == 'F3':
                f3s1 += 1
            if item[0] == 'F4':
                f4s1 += 1
            if item[0] == 'F5':
                f5s1 += 1

        elif item[1] == 'S2':
            s2 += 1
            if item[0] == 'F1':
                f1s2 += 1
            if item[0] == 'F2':
                f2s2 += 1
            if item[0] == 'F3':
                f3s2 += 1
            if item[0] == 'F4':
                f4s2 += 1
            if item[0] == 'F5':
                f5s2 += 1
```

```
        elif item[1] == 'S3':
            s3 += 1
            if item[0] == 'F1':
                f1s3 += 1
            if item[0] == 'F2':
                f2s3 += 1
            if item[0] == 'F3':
                f3s3 += 1
            if item[0] == 'F4':
                f4s3 += 1
            if item[0] == 'F5':
                f5s3 += 1

    print("a. P(F=F1|S=S1)=",f1s1,"/",s1,",","P(F=F1|S=S2)=",f1s2,"/",s2,",","P(F=F1|S=S3)=",f1s3,"/",s3,"\n"\
          "b. P(F=F2|S=S1)=",f2s1,"/",s1,",","P(F=F2|S=S2)=",f2s2,"/",s2,",","P(F=F2|S=S3)=",f2s3,"/",s3,"\n"\
          "c. P(F=F3|S=S1)=",f3s1,"/",s1,",","P(F=F3|S=S2)=",f3s2,"/",s2,",","P(F=F3|S=S3)=",f3s3,"/",s3,"\n"\
          "d. P(F=F4|S=S1)=",f4s1,"/",s1,",","P(F=F4|S=S2)=",f4s2,"/",s2,",","P(F=F4|S=S3)=",f4s3,"/",s3,"\n"\
          "e. P(F=F5|S=S1)=",f5s1,"/",s1,",","P(F=F5|S=S2)=",f4s2,"/",s2,",","P(F=F5|S=S3)=",f5s3,"/",s3,)


A = [['F1','S1'],['F2','S2'],['F3','S3'],['F1','S2'],['F2','S3'],['F3','S2'],['F2','S1'],['F4','S1'],['F4','S3']]

# Call the function to compute conditional probabilities
compute_conditional_probabilites(A)
```

```
a. P(F=F1|S=S1)= 1 / 4 , P(F=F1|S=S2)= 1 / 3 , P(F=F1|S=S3)= 0 / 3
b. P(F=F2|S=S1)= 1 / 4 , P(F=F2|S=S2)= 1 / 3 , P(F=F2|S=S3)= 1 / 3
c. P(F=F3|S=S1)= 0 / 4 , P(F=F3|S=S2)= 1 / 3 , P(F=F3|S=S3)= 1 / 3
d. P(F=F4|S=S1)= 1 / 4 , P(F=F4|S=S2)= 0 / 3 , P(F=F4|S=S3)= 1 / 3
e. P(F=F5|S=S1)= 1 / 4 , P(F=F5|S=S2)= 0 / 3 , P(F=F5|S=S3)= 0 / 3
```

Q9: Operations on sentences

You will be given two sentances S1, S2 your task is to find

a. Number of common words between S1, S2
b. Words in S1 but not in S2
c. Words in S2 but not in S1

Ex:

```
S1= "the first column F will contain only 5 unique values"
S2= "the second column S will contain only 3 unique values"
Output:
a. 7
b. ['first','F','5']
c. ['second','S','3']
```

In [9]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input strings
# you can free to change all these codes/structure

def string_features(S1, S2):
    """ This function compares two sentences S1 & S2 and find count of common words, words only in S1 and words
    """
    com = 0
    onlyS1, onlyS2 = [], []
    s1tokens, s2tokens = S1.split(),S2.split()

    #Loop for finding number of common words
    for word in s1tokens:
        if S2.find(word) != -1:
            com=com+1
        else:
            #Words only in S1
            onlyS1.append(word)

    #Loop to check words only in S2
    for word in s2tokens:
        if S1.find(word) == -1:
            onlyS2.append(word)

    return com, onlyS1, onlyS2

# sentences to check
S1= "the first column F will contain only 5 uniques values"
S2= "the second column S will contain only 3 uniques values"

#Call function
a,b,c = string_features(S1, S2)

#print values
print("a. ", a,"\nb. ",b,"\nc. ",c)
```

```
a.  7
b.  ['first', 'F', '5']
c.  ['second', 'S', '3']
```

Q10: Error Function

You will be given a list of lists, each sublist will be of length 2 i.e. [[x,y],[p,q],[l,m]..[r,s]] consider its like a martrix of n rows and two columns

a. the first column Y will contain interger values
b. the second column Y_{score} will be having float values
Your task is to find the value of $f(Y, Y_{score}) = -1 * \frac{1}{n}\Sigma_{foreachY,Y_{score}pair}(Ylog10(Y_{score}) + (1-Y)log10(1-Y_{score}))$ here n is the number of rows in the matrix

```
Ex:
[[1, 0.4], [0, 0.5], [0, 0.9], [0, 0.3], [0, 0.6], [1, 0.1], [1, 0.9], [1, 0.8]]
output:
0.44982
```

$$\frac{-1}{8} \cdot ((1 \cdot log_{10}(0.4) + 0 \cdot log_{10}(0.6)) + (0 \cdot log_{10}(0.5) + 1 \cdot log_{10}(0.5)) + \ldots + (1 \cdot log_{10}(0.8) + 0 \cdot log_{10}(0.2)))$$

In [10]:
```
# write your python code here
# you can take the above example as sample input for your program to test
# it should work for any general input try not to hard code for only given input strings
# you can free to change all these codes/structure

import math as m

def compute_log_loss(A):
    """ This function computes logloss for a given list of Y & Yscores.
    """

    logtrm = 0
    for lst in A:
        Y, Yscr = lst[0], lst[1]

        #calculate sum of log terms
        logtrm = logtrm + Y*m.log10(Yscr) + (1-Y)*(m.log10(1-Yscr))
    return -1*(logtrm/len(A))

A = [[1, 0.4], [0, 0.5], [0, 0.9], [0, 0.3], [0, 0.6], [1, 0.1], [1, 0.9], [1, 0.8]]

#call the function to compute log loss
compute_log_loss(A)
```

Out[10]:
```
0.42430993457031635
```