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]
             [23 30 37 44 51]]


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

referances

1. w3schools
2. geeks for geeks
3. program quiz
4. kaggle problems

In [16]:
```
# 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
# you can take matrix input from user or you can directly define the matrix and g
# reference for creating input - https://stackoverflow.com/questions/12293208/how

# you can free to change all these codes/structure
# here A and B are list of lists
def Multiply_multiplication(A,B):

    result=[ [0,0,0],    # here i m creating a lit of matix numbers of my variabl
             [0,0,0],
             [0,0,0] ]

    for i in range(len(A)):   # this is range for row

        for j in range(len(B[0])):# this is range for column

            for k in range(len(B)):
                result[i][j] += A[i][k] * B[k][j]

    for p in result:
        print(p)



A = [ [1, 2, 3],
      [6, 7, 4],
      [8, 10, 11] ]

B = [[1, 5, 3],
     [2, 6, 5],
     [7, 4, 9] ]

print("Result of matrix multiplication of A and B is : ")
Multiply_multiplication(A,B)
```

```
Result of matrix multiplication of A and B is :
[26, 29, 40]
[48, 88, 89]
[105, 144, 173]
```

Q2: Proportional Sampling - 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

#video link for the topic - https://www.appliedaicourse.com/lecture/11/applied-mc
# you can free to change all these codes/structure


def pick_randomvalue_fromlist(A):
    sum = 0   # here i m creating a empty variable sum to save my output
    cum_sum = []   # here i m intiating a cum_sum from emplty list

    for K in range (len(A)):
        sum = sum + A[K]
        cum_sum.append(sum)  # here we will add the sum vlaue in cum_sum to get t


    r = int (random.uniform(0, sum ))  # here i m creating the random number
    print(r)
    number =0
    for index in range (len(cum_sum)):
        if (r >= cum_sum[index]  and r < cum_sum[index+1]):
            return A[index+1]
        return number

def sample_magnitude():
    A = (1, 5, 27, 6, 13, 28, 100, 45, 10, 79)
    a = dict()    # here i have created the empty dictionary

    print(A, sum(A))

    for K in range(1,100):
        number = pick_randomvalue_fromlist(A)   # here we will pick the random nu
        if number not in a:
            a[number] = 1
        else:
            a[number]+=1
    print(a)



sample_magnitude()   # now we will call the our output of sample magnitude
```

```
(1, 5, 27, 6, 13, 28, 100, 45, 10, 79) 314
312
295
135
199
181
186
246
34
24
```

```
118
16
112
224
274
109
0
124
8
286
109
169
136
66
284
16
271
25
278
159
154
220
61
275
115
24
34
292
28
250
26
36
261
313
102
238
43
46
174
169
234
299
72
163
100
182
1
57
12
120
106
226
90
306
118
200
237
```

```
18
269
99
195
122
266
64
102
239
4
257
57
188
223
231
71
170
283
176
76
255
225
201
72
267
38
244
39
37
158
30
11
41
{0: 97, 5: 2}
```

Q3: Replace the digits in the string with

consider a string that will have digits in that, we need to remove all the 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 [11]:
```
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
# try to complete this question using regular expressions
# you can free to change all these codes/structure
# String: it will be the input to your program

my_string = " my Name is Anuj and i m 21 year old  .and i enrolled in applied ai

print("my original string :" + str(my_string))

S = "#"

for ele in my_string:
    if ele.isdigit():
        my_string =my_string.replace(ele, S)

print("------**********----------**********----------*********")

print("my new string is :" + str(my_string))
```

```
my original string : my Name is Anuj and i m 21 year old  .and i enrolled in ap
plied ai course in 2022 and my phone no is 115544885545
------**********----------**********----------**********----
---------***
my new string is : my Name is Anuj and i m ## year old  .and i enrolled in appl
ied ai course in #### and my phone no is ############
```

Q4: Students marks dashboard

consider the marks list of class students given two lists
Students =
['student1','student2','student3','student4','student5','student6','student7','student8','student9','student
Marks = [45, 78, 12, 14, 48, 43, 45, 98, 22, 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','studen
t6','student7','student8','student9','student10']
Marks = [45, 78, 12, 14, 48, 43, 47, 98, 22, 80]
a.
student8  98
student10 80
student2  78
student5  48
student7  47
b.
student3 12
student4 14
student9 22
student6 43
student1 45
c.
student9 22
student6 43
student1 45
student7 47
student5 48
```

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


Students=['student1','student2','student3','student4','student5','student6','stud
Marks = [45, 78, 12, 14, 48, 43, 47, 98, 22, 80]

grades =list(zip(Students, Marks))

grades.sort(key = lambda e : e[1])

# you can free to change all these codes/structure
def display_dash_board(students, marks):
    # write code for computing top top 5 students
    top_5 = grades[:5]
    top = top_5[::-1]
    top_5_student= top

    # write code for computing top least 5 students
    least = grades[:5]
    least_5_students = least # compute this


    # write code for computing top least 5 students
    twenty_fifth = math.ceil(len(grades)//4)
    seventy_fifth = math.floor(3 *(len(grades)//4))
    middle = grades [twenty_fifth : seventy_fifth]
    students_within_25_and_75 =middile  # compute this

    return top_5_students, least_5_students, students_within_25_and_75

    top_5_students, least_5_students, students_within_25_and_75 = display_dash_bo
print(top_5_students)
print(least_5_students)
print( students_within_25_and_75)
```

```
[('student8', 98), ('student10', 80), ('student2', 78), ('student5', 48), ('stu
dent7', 47)]
[('student3', 12), ('student4', 14), ('student9', 35), ('student6', 43), ('stud
ent1', 45)]
[('student9', 35), ('student6', 43), ('student1', 45), ('student7', 47), ('stud
ent5', 48)]
```

Q5: Find the closest points

Consider you have 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 defind 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)
```

Hint - If you write the formula correctly you'll get the distance between points (6,-7) and (3,-4) = 0.065

In [6]:

```
# 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
# you can free to change all these codes/structure


# here S is list of tuples and P is a tuple ot len=2

import math
def closest_point(S, P):
    closest_point = []                      # here i m creating empty list to saj
    my_final_list_of_points = []

     # here i m writing my varibale dinomirator and nomirator according to formul

    for point in S:
        dinomirator = math.sqrt((point[0] ** 2) + (point[1] ** 2)) * math.sqrt((P
        nomirator = point[0] * P[0] + point[1] * P[1]

        if dinomirator != 0:       # here my main case of execuation of my formul
            cosine_distance_for_this_point = math.acos(nomirator / dinomirator)
            closest_point.append((cosine_distance_for_this_point, point))

    for item in sorted(closest_point, key=lambda x: x[0])[:5]:        # here i m
         my_final_list_of_points .append(item[1])

    return  my_final_list_of_points


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

closest_point = closest_point(S, P)
print("Closest point-cosine-distance - top 5:", *[point for point in  closest_poi
```

```
Closest point-cosine-distance - top 5:
(6, -7)
(1, -1)
(6, 0)
(-5, -8)
(-1, -1)
```

Q6: Find Which line separates oranges and apples

consider you have 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 formate, i.e list of strings)

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

your task is to for each line that is given print "YES"/"NO", you will print yes, if all the red points are one side of the line and blue points are other side of the line, otherwise 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



def find_which_line_seperate(red, blue, line):
    red_point_sign = -1

    if eval(line.replace('x', '*%s' % red[0][0]).replace('y', '*%s' % red[0][1])
        red_point_sign  = 1

        # in the above we checking the contidition by f statement that if value i

    for r_pt in red:
        if red_point_sign  == 1 and eval(
                line.replace('x', '*%s' % r_pt[0]).replace('y', '*%s' % r_pt[1])
            return 'NO'
        # here we are checking that if the red point sign is  then vlaue is less


        if red_point_sign  == -1 and eval(
                line.replace('x', '*%s' % r_pt[0]).replace('y', '*%s' % r_pt[1])
            return 'NO'
        # in this line we are checking when value of red point is - and value of


    blue_point_sign = -1 * red_point_sign # now here we are executing our formula

    for b_pts in blue:
        if blue_point_sign == 1 and eval(
                line.replace('x', '*%s' % b_pts[0]).replace('y', '*%s' % b_pts[1]
            return 'NO'

        if blue_point_sign == -1 and eval(
                line.replace('x', '*%s' % b_pts[0]).replace('y', '*%s' % b_pts[1]
            return 'NO'

    return 'YES'

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 = find_which_line_seperate(Red, Blue, i)
    print(yes_or_no)
```

```
YES
NO
NO
YES
```

Q7: Filling the missing values in the specified formate

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 th
e 24 equally to all 4 places

Ex 2: 40, _, _, _, 60 ==> (60+40)/5,(60+40)/5,(60+40)/5,(60+40)/5,(60+4
0)/5 ==> 20, 20, 20, 20, 20 i.e. the sum of (60+40) is distributed quall
y to all 5 places

Ex 3: 80, _, _, _, _  ==> 80/5,80/5,80/5,80/5,80/5 ==> 16, 16, 16, 16, 1
6 i.e. the 80 is distributed qually to all 5 missing values that are rig
ht 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, 1
0, 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, 1
2, 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 [17]:
```
# 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
#run your code in the function for each of the inputs mentioned above and make su

# you can free to change all these codes/structure
def curve_smoothing(string):
    index_of_cell_list  = []   # here i m creating a empty list to safe the repea

    split_string = string.split(',')  # here i m spliting a my varible for given

    for index in range(len(split_string)):
        if split_string[index] != '_':
             index_of_cell_list.append(index)  # here i m adding varible index to


    index_of_cell_list.append(len(split_string) - 1)

    print( index_of_cell_list)
    [2, 6, 8,]


    start = 0   # here i m creating a varible start from 0 to safe the my furtue

    for element in  index_of_cell_list:

        cumulative_sum_prev_and_next_value = int(split_string[element]) if split_
        cumulative_sum_prev_and_next_value += int(split_string[start]) if split_s

      # this above line code snippet taken from stack overflow

        integer_to_replace_each_previous_empty_cell = cumulative_sum_prev_and_nex
                element - start + 1)


        split_string = [integer_to_replace_each_previous_empty_cell if start <= x
            in range(len(split_string))]


        start = element

    return split_string


S = "_,_,30,_,_,_,50,_,_"
result = smoothed_values = curve_smoothing(S)

print('**********************************************************************

print(" curve smoothing values are :")
print(result)
```

```
[2, 6, 8]
**********************************************************************
```

```
 curve smoothing values are :
[10, 10, 12, 12, 12, 12, 4, 4, 4]
```

Q8: Filling the missing values in the specified formate

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,S
3],[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 [29]:
```
# 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
# you can use nested loops or dictionaries to write your code


# you can free to change all these codes/structure
def compute_conditional_probabilites(A,B ):
    dinomirator = 0    # here i m intiating the dinomirator from 0
    numerator =0        # here i m intiating the numerator from 0

    for i in range (len(Z)):
        if (Z [i][1]== B):
            dinomirator = dinomirator +1
            if (Z[i][0]== A):
                numerator = numerator +1

    print (" p ( A == {} | B == {} ) ={}/{} ".format(A, B,  str(numerator), str(d

Z = [['F1','S1'],['F2','S2'],['F3','S3'],['F1','S2'],['F2','S3'],['F3','S2'],['F2

for i in  ["F1","F2","F3","F4","F5"]:  # lets call the output from given range
    for j in ["S1","S2", "S3"]:
        compute_conditional_probabilites(i,j)
```

```
 p ( A == F1 | B == S1 ) =1/4  p ( A == F1 | B == S2 ) =1/3  p ( A == F1 | B ==
S3 ) =0/3  p ( A == F2 | B == S1 ) =1/4  p ( A == F2 | B == S2 ) =1/3  p ( A ==
F2 | B == S3 ) =1/3  p ( A == F3 | B == S1 ) =0/4  p ( A == F3 | B == S2 ) =1/3
p ( A == F3 | B == S3 ) =1/3  p ( A == F4 | B == S1 ) =1/4  p ( A == F4 | B ==
S2 ) =0/3  p ( A == F4 | B == S3 ) =1/3  p ( A == F5 | B == S1 ) =1/4  p ( A ==
F5 | B == S2 ) =0/3  p ( A == F5 | B == S3 ) =0/3
```

Q9: Given two sentances S1, S2

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

In [19]:
```
# 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

# you can free to change all these codes/structure
def string_features(S1, S2):

    S1_words = set(S1.split())                        # here i m creating two vari
    S2_words = set(S2.split())

    M = len( S1_words & S2_words)   # here i using & because i want common words

    N= set(S1_words) - (set(S2_words))
    list_of_N = list(N)

    O = set(S2_words) -(set(S1_words))
    list_of_O = list(O)

    return M, list_of_N, list_of_O

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


M,list_of_N,list_of_O = string_features(S1, S2)

print(M)
print(list_of_N)
print(list_of_O)
```

```
7
['F', 'first', '5']
['3', 'second', 'S']
```

Q10: Given two sentances S1, S2

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:
0.4243099
```

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

In [26]:
```
# 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


# you can free to change all these codes/structure

from math import log

def compute_log_loss(A):
    loss = 0   # at the starting i m intiating my loss variable from 0 to safe th
    for row in A:
        loss += (row[0]* log(row[1],10)) + ((1- row[0])* log( 1- row[1],10))  # h
        log_loss = - 1 * loss / len (A)
    return log_loss




A = [[1, 0.4], [0, 0.5], [0, 0.9], [0, 0.3], [0, 0.6], [1, 0.1], [1, 0.9], [1, 0.
loss = compute_log_loss(A)
print(loss)
```

```
0.42430993457031635
```

In [ ]: