**Problem Statement: The Doomed Dice Challenge**

**Part B**

Code....

def prob(DieA,DieB):

Mat=[]

uni=[]

D={}

for i in DieA:

for j in DieB:

total=i+j

Mat.append(total)

if total not in uni:

uni.append(total)

uni.sort()

for i in uni:

D.update({i:(Mat.count(i)/36)})

return D

def all\_com():

listD = []

for b in range(2, 9):

for c in range(b, 9):

for d in range(c, 9):

for e in range(d, 9):

for f in range(e, 9):

dice = [1, b, c, d, e, f]

listD.append(dice)

return listD

def undoom\_dice(DieA,DieB):

org=prob(DieA,DieB)

L=[]

for i in all\_com():

H=0

for j in i:

if j<=4:

H=H+1

else:

break

if H==6:

L.append(i)

for i in L:

for j in all\_com():

D1=prob(i,j)

if org==D1:

return (i,j)

break

Die\_A=[1,2,3,4,5,6]

Die\_B=[1,2,3,4,5,6]

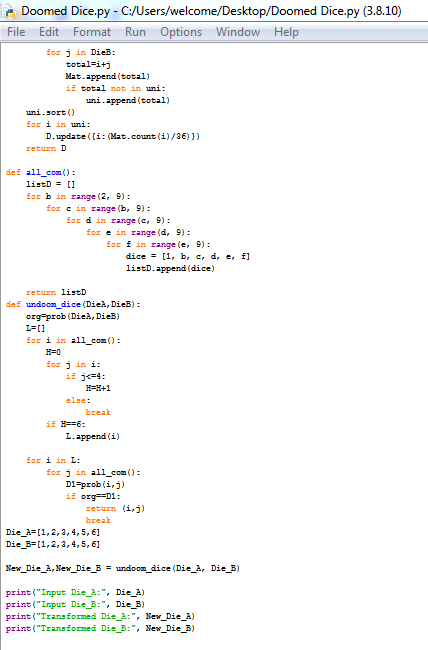
New\_Die\_A,New\_Die\_B = undoom\_dice(Die\_A, Die\_B)

print("Input Die\_A:", Die\_A)

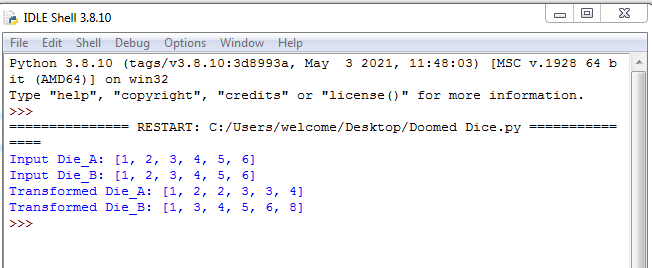
print("Input Die\_B:", Die\_B)

print("Transformed Die\_A:", New\_Die\_A)

print("Transformed Die\_B:", New\_Die\_B)



Output

****

Logical Explanatio

**prob(DieA, DieB)` function**

This function takes two lists `DieA` and `DieB`, representing the faces of two dice.

It calculates the sum of all possible pairs of faces from `DieA` and `DieB` and stores them in the list `Mat`.

It creates a unique list `uni` containing all unique sums obtained from the dice pairs.

Finally, it calculates the probability of each sum and stores it in a dictionary `D`. The probability is calculated by dividing the count of each sum in `Mat` by the total number of possible combinations (36 in this case, as both dice are six-sided).

**`all\_com()` function**

This function generates all possible combinations of six-sided dice faces.

It uses nested loops to create combinations where the faces are in non-decreasing order.

`**undoom\_dice(DieA, DieB)` function**

This function takes two lists `DieA` and `DieB` and calculates the probability distribution using the `prob` function.

It then iterates through all possible combinations of dice faces generated by the `all\_com` function.

For each combination, it checks if all faces are less than or equal to If so, it appends that combination to the list `L`.

Next, it iterates through all possible combinations again and calculates the probability distribution for each pair using the `prob` function.

If the calculated probability distribution matches the original distribution (`org`), it returns the pair of dice faces as a tuple `(i, j)` and breaks out of the loop.

**Main Function**

It defines two six-sided dice (`Die\_A` and `Die\_B`) with faces 1 to 6.

It calls the `undoom\_dice` function with these two dice.

It prints the original and transformed dice faces.