

Part A:

1. How many total combinations are possible? Show the math along with the code!

Die_A=[1,2,3,4,5,6]

Die_B=[1,2,3,4,5,6]

Total possible combinations:

(1,1), (1,2), (1,3), (1,4), (1,5), (1,6),
(2,1), (2,2), (2,3), (2,4), (2,5), (2,6),
(3,1), (3,2), (3,3), (3,4), (3,5), (3,6),
(4,1), (4,2), (4,3), (4,4), (4,5), (4,6),
(5,1), (5,2), (5,3), (5,4), (5,5), (5,6),
(6,1), (6,2), (6,3), (6,4), (6,5), (6,6)

Therefore, total combinations are 36

The screenshot shows a Python IDE with a code editor on the left and an output panel on the right. The code in the editor is as follows:

```
1 faces_on_die_a = 6
2 faces_on_die_b = 6
3 total_combinations = faces_on_die_a * faces_on_die_b
4 print("Total Combinations = "+str(total_combinations))
5
```

The output panel shows the result of running the code:

```
*****
Total Combinations = 36
```

At the bottom of the code editor, there is a dropdown menu set to "Python 3.10" and two buttons: "SAVE" and "+ Create New".

2. Calculate and display the distribution of all possible combinations that can be obtained when rolling both Die A and Die B together. Show the math along with the code!

(1,1), (1,2), (1,3), (1,4), (1,5), (1,6) = [2,3,4,5,6,7]
(2,1), (2,2), (2,3), (2,4), (2,5), (2,6) = [3,4,5,6,7,8]
(3,1), (3,2), (3,3), (3,4), (3,5), (3,6) = [4,5,6,7,8,9]
(4,1), (4,2), (4,3), (4,4), (4,5), (4,6) = [5,6,7,8,9,10]
(5,1), (5,2), (5,3), (5,4), (5,5), (5,6) = [6,7,8,9,10,11]
(6,1), (6,2), (6,3), (6,4), (6,5), (6,6) = [7,8,9,10,11,12]

```

1 faces_on_die_a = 6
2 faces_on_die_b = 6
3 distribution_matrix = []
4
5 for i in range(1,faces_on_die_a+1):
6     arr=[]
7     for j in range(1,faces_on_die_b+1):
8         arr.append(i+j)
9     distribution_matrix.append(arr)
10
11 for row in distribution_matrix:
12     print(row)
13

```

Python 3.10

SAVE

+ Create New

INPUT

DEBUG

RUN

OUTPUT

```

[2, 3, 4, 5, 6, 7]
[3, 4, 5, 6, 7, 8]
[4, 5, 6, 7, 8, 9]
[5, 6, 7, 8, 9, 10]
[6, 7, 8, 9, 10, 11]
[7, 8, 9, 10, 11, 12]

```

3. Calculate the Probability of all Possible Sums occurring among the number of combinations from (2).

$P(\text{event}) = \text{favourable chances} / \text{total number of chances}$

For $P(\text{sum}=2)$, (1,1) is the only possibility

Therefore $P(\text{sum}=2) = 1/36$

For $P(\text{sum}=3)$, (1,2) & (2,1) are the possibilities

Therefore $P(\text{sum}=3) = 2/36$ like we have to calculate the remaining probabilities up to $P(\text{sum}=12)$

```

1 faces_on_die_a = 6
2 faces_on_die_b = 6
3 distribution_matrix = []
4 for i in range(1,faces_on_die_a+1):
5     arr=[]
6     for j in range(1,faces_on_die_b+1):
7         arr.append(i+j)
8     distribution_matrix.append(arr)
9
10 for i in range(2,13):
11     count=0
12     for j in distribution_matrix:
13         if(i in j):
14             count+=1
15     print("P(Sum="+str(i)+"") +str(round(count/36,2)))
16

```

Python 3.10

SAVE

+ Create New

INPUT

DEBUG

RUN

OUTPUT

```

P(Sum=2)=0.03
P(Sum=3)=0.06
P(Sum=4)=0.08
P(Sum=5)=0.11
P(Sum=6)=0.14
P(Sum=7)=0.17
P(Sum=8)=0.14
P(Sum=9)=0.11
P(Sum=10)=0.08
P(Sum=11)=0.06
P(Sum=12)=0.03

```

Part B:

- Die A cannot have more than 4 Spots on a face.
- Die A may have multiple faces with the same number of spots.
- Die B can have as many spots on a face as necessary i.e. even more than 6.

Based on the above conditions the possible combinations for Die A is :

[1,1,1,1,1,1],

[1,1,1,1,1,2],

.

.

.

[4,4,4,4,4,4]

Possible combinations for Die B is

[1,1,1,1,1,1],

[1,1,1,1,1,2],

.

.

.

[8,8,8,8,8,8]

The maximum value in Die B is 8 , because the maximum value in Die A is 4

CODE:

```
def get_sums(die1, die2):  
    sums = []  
    for i in die1:  
        for j in die2:  
            sums.append(i+j)  
    return sums
```

```
def get_freqs(sums):  
    freqs = { }  
    for s in sums:  
        if s not in freqs:  
            freqs[s] = 0  
        freqs[s] += 1  
    return freqs
```

```
def calc_probs(freqs, n):  
    probs = { }
```

```

    for s, f in freqs.items():
        probs[s] = f / n
    return probs

def dice_match(die1, die2, orig_probs):
    sums = get_sums(die1, die2)
    freqs = get_freqs(sums)
    probs = calc_probs(freqs, len(die1)*len(die2))
    return probs == orig_probs

def transform_dice(die1, die2):
    orig_probs = calc_probs(get_freqs(get_sums(die1, die2)), 36)

    for a in [1,2,3,4]:
        for b in [1,2,3,4]:
            for c in [1,2,3,4]:
                for d in [1,2,3,4]:
                    for e in [1,2,3,4]:
                        for f in [1,2,3,4]:
                            new_die1 = [a,b,c,d,e,f]
                            for i in range(1, 9):
                                for j in range(1, 9):
                                    for k in range(1,9):
                                        for l in range(1,9):
                                            for m in range(1,9):
                                                for n in range(1,9):
                                                    new_die2=[i,j,k,l,m,n]
                                                    if dice_match(new_die1, new_die2, orig_probs):
                                                        return new_die1, new_die2

    print("No solution found")

die1 = [1, 2, 3, 4, 5, 6]
die2 = [1, 2, 3, 4, 5, 6]

new_die1, new_die2 = transform_dice(die1, die2)

print(new_die1)
print(new_die2)

```

OUTPUT:

Die A=[1, 2, 2, 3, 3, 4]
 Die B=[1, 3, 4, 5, 6, 8]

OPTIMIZED CODE:

```

def get_sums(die1, die2):
    sums = []
    for i in die1:
        for j in die2:
            sums.append(i+j)
    return sums

```

```

def transform_dice(die1, die2):
    res_sum=get_sums(die1,die2)
    res_sum.sort()

    for a in [1,2,3,4]:
        for b in [1,2,3,4]:
            for c in [1,2,3,4]:
                for d in [1,2,3,4]:
                    for e in [1,2,3,4]:
                        for f in [1,2,3,4]:
                            new_die1 = [a,b,c,d,e,f]
                            for i in range(1, 9):
                                for j in range(1, 9):
                                    for k in range(1,9):
                                        for l in range(1,9):
                                            for m in range(1,9):
                                                for n in range(1,9):
                                                    new_die2=[i,j,k,l,m,n]
                                                    if get_sums(new_die1,new_die2).sort()==res_sum:
                                                        return new_die1, new_die2

    print("No solution found")

die1 = [1, 2, 3, 4, 5, 6]
die2 = [1, 2, 3, 4, 5, 6]

new_die1, new_die2 = transform_dice(die1, die2)

print(new_die1)
print(new_die2)

```

OUTPUT:

Die A=[1, 2, 2, 3, 3, 4]
Die B=[1, 3, 4, 5, 6, 8]