**PART A:**

**[1]**

The combination of possible outcomes for one die will always be the number faces to

the power of the outcomes we need. So here it will be 6 to the power 1 for one die.

Therefore, to get the total number of outcomes when two die is rolled together would

just be (6^1\*6^1) where we just add the outcomes of the two dice.

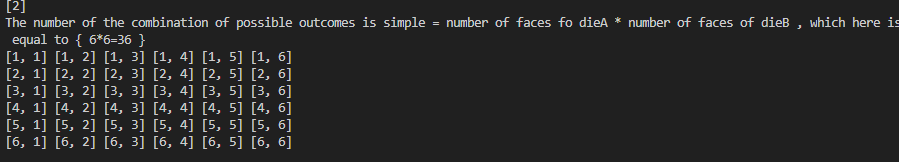
Here equal to **6\*6 = 36.**

**[2]**

The number of the combination of possible outcomes is

simply = number of faces of dieA \* number of faces of dieB , which here is equal to

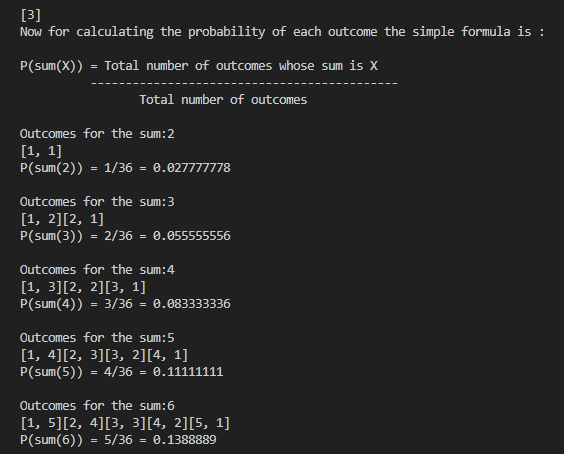
**{ 6\*6=36 }**

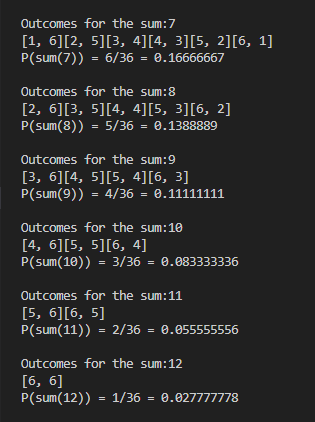


[3]

Now for calculating the probability of each outcome the simple formula is :

P(sum(X)) = Total number of outcomes whose sum is X / Total number of outcomes





**PART B:**

The only idea I got in solving this is using brute force calculate each possible die combination

of A and B and then calculating the probability to see whether the probability is same as

initial undoomed dice.

But it is not at all a good solution as it might contains huge number of combinations and

the time complexity is worst.

The only optimation we can use is that to find from where we can start the doing the brute-force.

we know that the least/min sum we need is 2 =(1+1) is the only combination we can use to get this.

so it is visible that we should have 1 in each die array.

so the start is :

dieA = [1,?,?,?,?,?]

dieB = [1,?,?,?,?,?]

dieA can start from [1,1,1,1,1,,] and

dieB can start from [1,2,3,4,5,6] as it cannot have duplicate and that this is only the

minimum array possible for dieB.

now we need to always have one pair in both die that sums upto to 12.

Now we can start calculating each combination of dieA for each of dieB and then calculating

the probability if it matches the initial.

I have not done the coding...