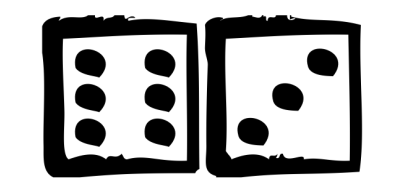
**Securin - The Doomed Dice Challenge**

**Problem Statement:**

You are given two six-sided dice, Die A and Die B, each with faces numbered from 1 to 6. You can only roll both the dice together & your turn is guided by the obtained sum. Example: Die A = 6, Die B = 3. Sum = 6 + 3 = 9



You may represent Dice as an Array or Array-like structure. Die A = [1, 2, 3, 4, 5, 6] where the indices represent the 6 faces of the die & the value on each face.

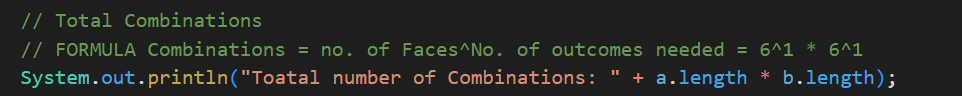
**Part-A**

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

The possible combinations are calculated with the total number of possibilities of Die A and Die B in which there are 36 distinct and possible ways for the dice to come up when rolled. The number of ways the first die can come up 6 {1, 2, 3, 4, 5, 6} is multiplied by the number of ways the second die can come up 6 {1, 2, 3, 4, 5, 6}. Here, the **combination for single die = no. of face \* no. of outcomes.**

**Formula**: Total Possibilities = **6^1 \* 6^1 = 36**.

**CODE:**

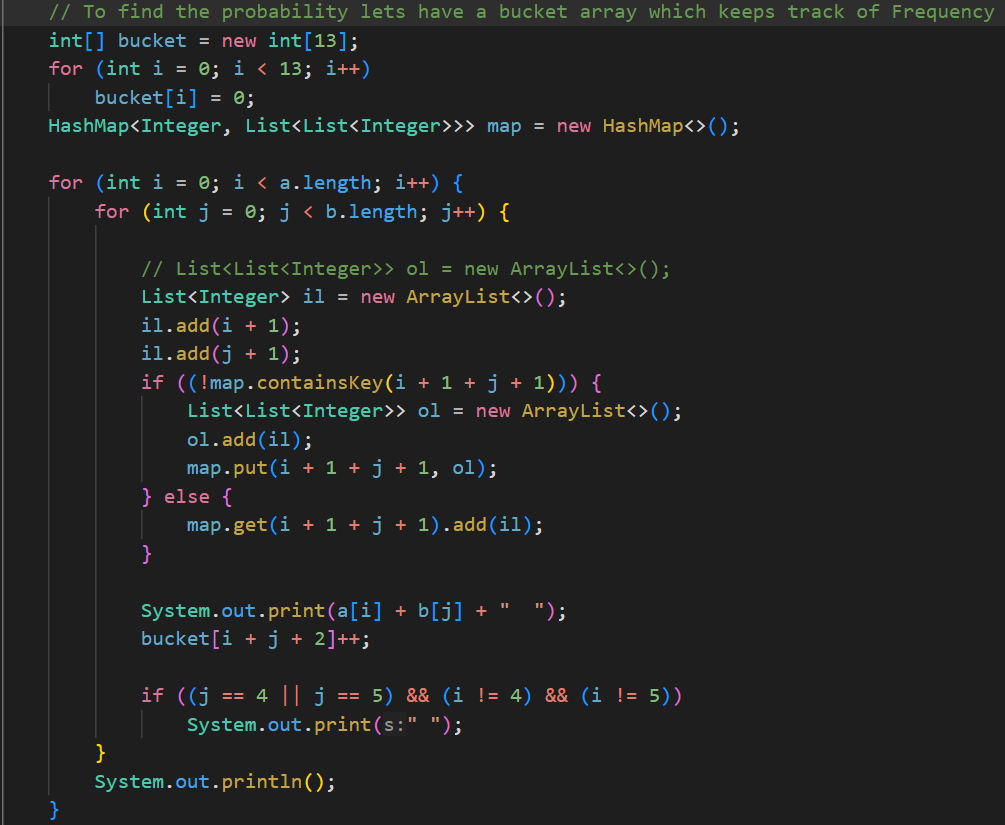


**OUTPUT:**

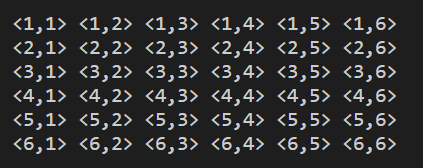


***2. Calculate and display the distribution of all possible combinations that can be obtained when rolling both Die A and Die B together. Show the math along with the code! [ Hint: A 6 x 6 Matrix. ]***

As, discussed above the Total no. of possible combinations are 36.

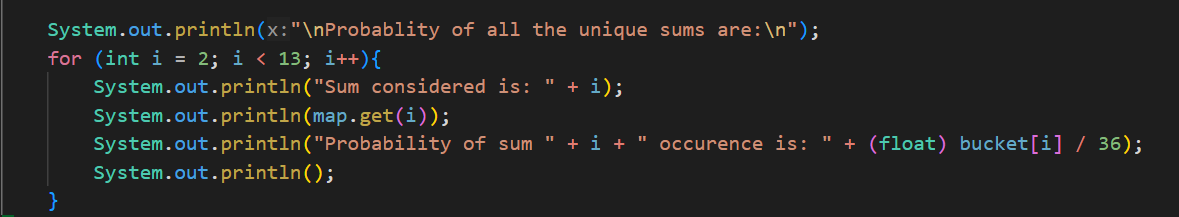


**OUTPUT:**

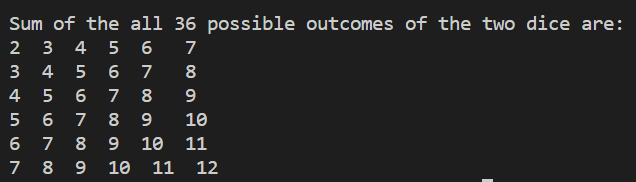


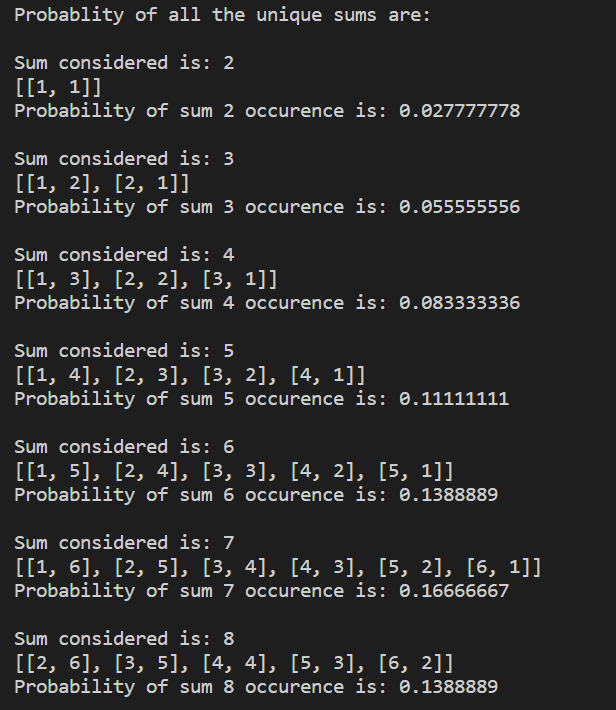
***3. . Calculate the Probability of all Possible Sums occurring among the number of combinations from (2). Example: P(Sum = 2) = 1/X as there is only one combination possible to obtain Sum = 2. Die A = Die B = 1.***

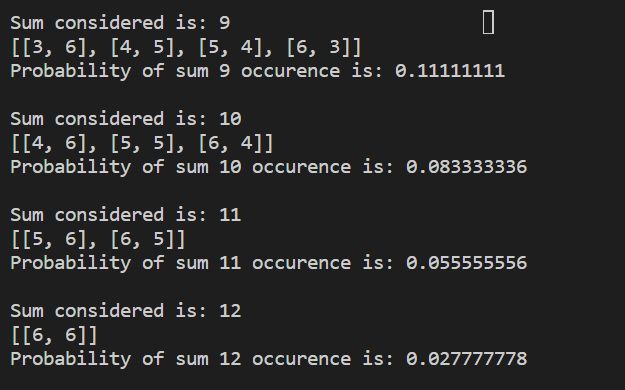
**CODE:**



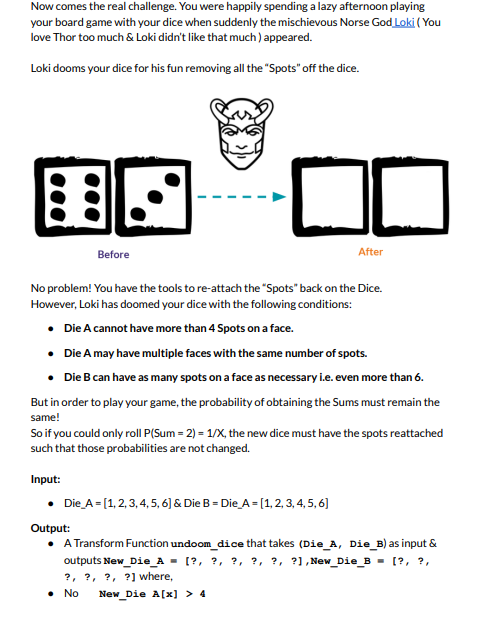
**OUTPUT:**





****

**PART B**



**EXPLANATION:**

There are various approaches to this problem, basically the possible outcomes that are possible when the constraints of dice,

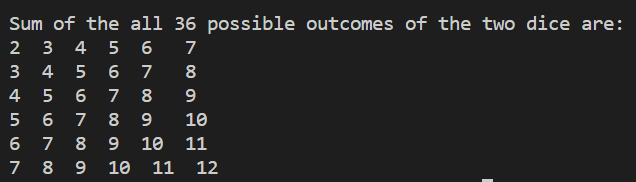
Die A = { 1, 2, 3, 4 }

Die B = { 1, 2, 3, 4, 5, ….}

🡪 Constraint (i.e) max no. of dots that can be present on a single face

But,

P (Sum of the Die A + Die B ) must be same as before original die (i.e)



This should be achieved even after the change of dots on each face of the die given the order not considered.

APPROACH 1: Brute Force

APPROACH 2: Recursion

Since both approaches almost go with the same complexity I go with the Brute Force Approach. In which every possible combination is considered and checked if it satisfies the given condition. But this will lead to a higher Time complexity. So, to optimize this the number of combinations considered is tried to be reduced.

Min sum = 2 [1+1] which leaves us with 5 left combinations (i.e),

Die A = { 1, \_, \_, \_, \_, \_ } Die B = { 1, \_, \_, \_, \_, \_ }

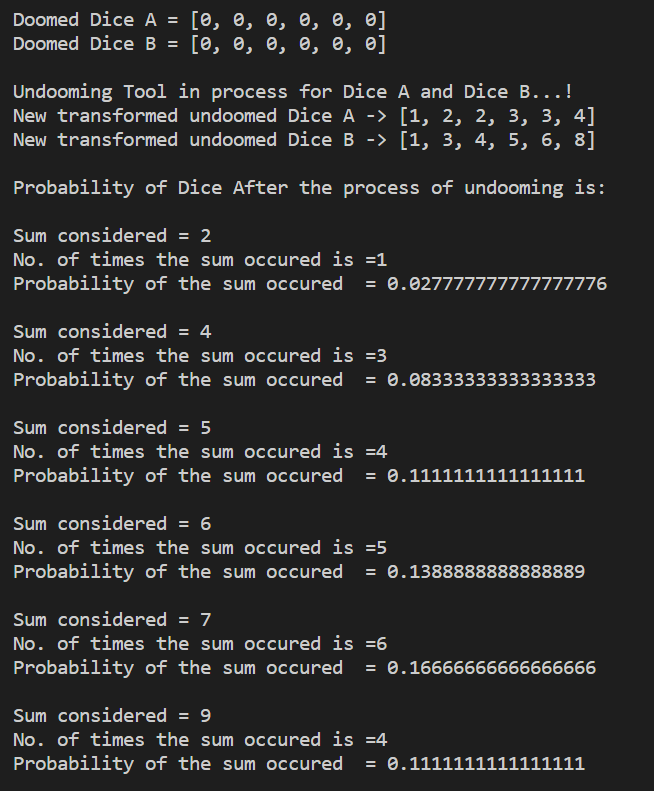
And, Die B cannot have repetitions but Die A can (i.e), they start from

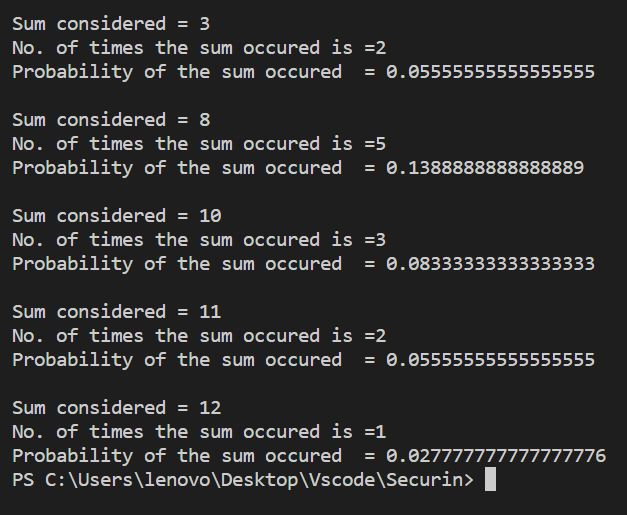
Die A = { 1, 1, 1, 1, 1, 1 } and Die B = { 1, 2, 3, 4, 5, 6 }

This will help us reduce the unwanted execution of combinations in the loop.

* Consider you have a combination that has the probability distribution the same as the original one but fails to have a dice combination that gives the sum 12 then it is excluded.
* So, it is better to check if there exists a sum 12 prior to the checking process execution of the other sum probabilities

**OUTPUT:**

****



----------------------------------------------------------------