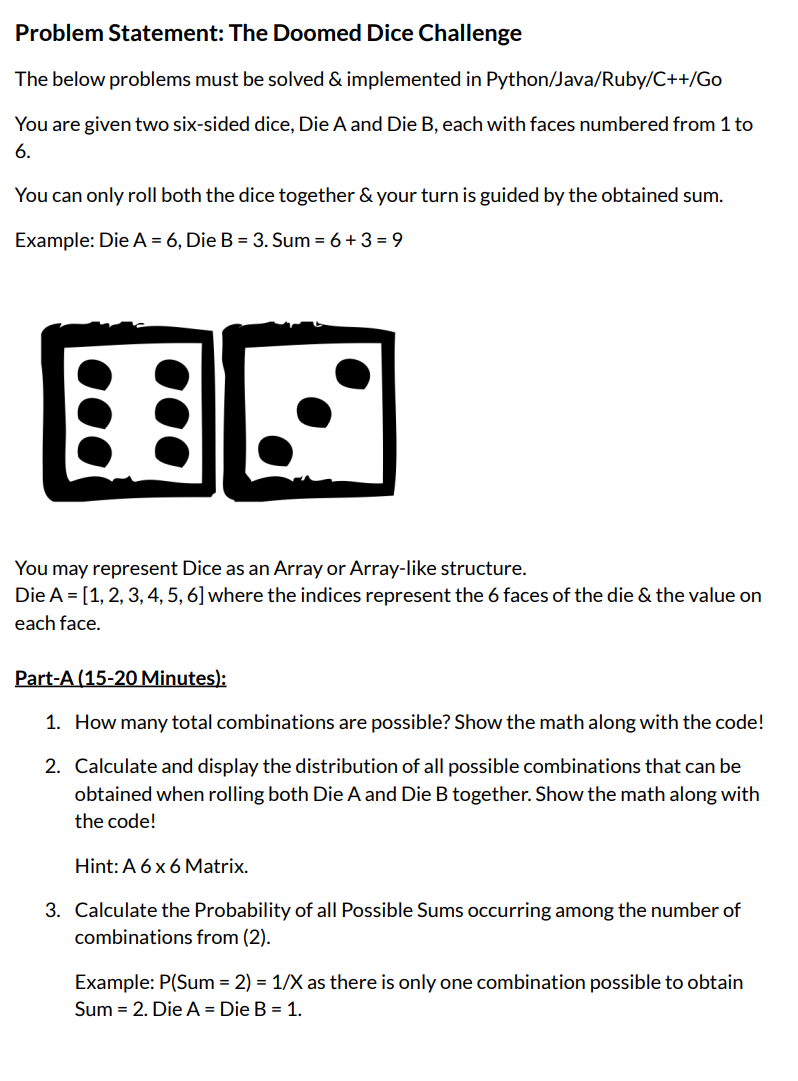
SOLUTION TO DICE PROBLEM:



1. **Total Combinations of dice rolls possible: 36**

(1,1),(1,2),I1,3),(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).

1. **All possible sums that can be obtained :**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sums | 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 5 | 6 | 7 | 8 | 9 | 10 | 11 |
| 6 | 7 | 8 | 9 | 10 | 11 | 12 |

**Function to find the sums:**

public static int[][] findcombinations(int combination\_distribution[][]){

        for(int i=0;i<6;i++){

            for(int j=0;j<6;j++){

                combination\_distribution[i][j]=(i+1)+(j+1);

            }

        }

        return combination\_distribution;

    }

3.**Probability of all sums occurring among the number of combinations:**

//logic

Probability= Number of occurrences of a sum/ total possibilities;

Probability of 2= 1/36

Probability of 3=2/36

Probability of 4=3/36

Probability of 5=4/36

Probability of 6=5/36

Probability of 7 =6/36

Probability of 8 = 5/36

Probability of 9= 4/36

Probability of 10=3/36

Probability of 11=2/36

public static double[] probability(int total\_combinations,int[][] combination\_distribution){

        int min\_sum=2; //1+1; when dice 1 is 1 and dice 2 is 1

        int max\_sum=12; //6+6 when dice 1 is 6 and dice 2 is 6

        double[] dp=new double[13];// frequency matrix

        double total=total\_combinations;

            for(int i=0;i<combination\_distribution.length;i++){

                for(int j=0;j<combination\_distribution[0].length;j++){

                    dp[combination\_distribution[i][j]]++;

                }

            }

            System.out.println("Probablilites for sums :");

            for(int i=min\_sum;i<=max\_sum;i++){

                  double prob=dp[i]/total;

                  System.out.println(i+" : "+dp[i]+" / "+total+"  = "+prob);

            }

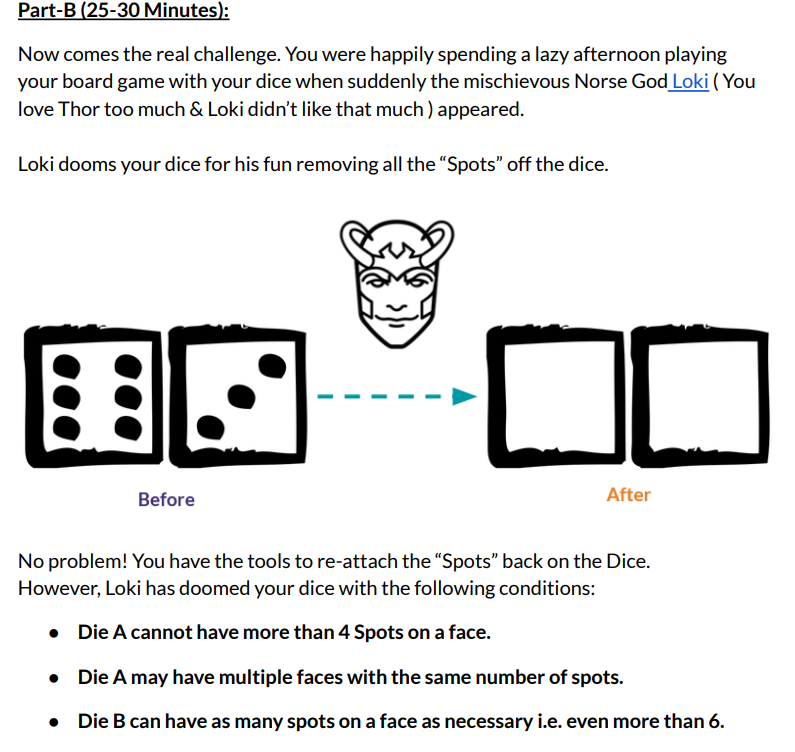
            return dp;

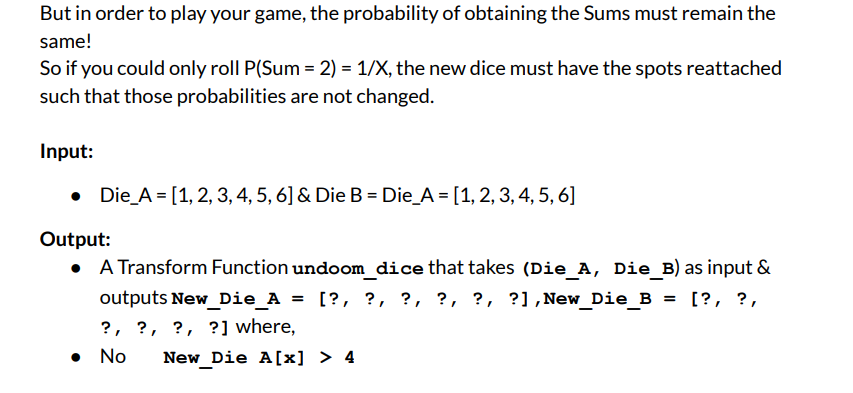
        }

Probability of 12=1/36

**Function to find Probabilities:**

1. Create a frequency array that stores the number of occurrences of each sum.
2. Traverse through the frequency array and find the probabilities using the above formula





**LOGIC:**

**Step1:**

This can be solved by generating all arrays of size 6 within {1,2,3,4} and with repetition. And all possible arrays of size 6 within{1,2,3,4,5,6,7,8,9,10,11} without repetition.

**static List<int[]> generateDiceACombinations() {**

**List<int[]> combos = new ArrayList<>();**

**generateDiceA(new int[6], 0, combos);**

**return combos;**

**}**

**static void generateDiceA(int[] curr, int index, List<int[]> combos) {**

**if (index == 6) {**

**combos.add(curr.clone());**

**return;**

**}**

**if (index == 0 ) {**

**curr[index] = 1;**

**generateDiceA(curr, index + 1, combos);**

**}**

**else if(index == 1){**

**curr[index] = 4;**

**generateDiceA(curr, index + 1, combos);}**

**else {**

**curr[index] = 2;**

**generateDiceA(curr, index + 1, combos);**

**curr[index] = 3;**

**generateDiceA(curr, index + 1, combos);**

**}**

**}**

**static void generateDiceB(int[] curr, int index, List<int[]> combos) {**

**if (index == 6) {**

**combos.add(curr.clone());**

**return;**

**}**

**if (index == 0) {**

**curr[index] = 1;**

**generateDiceB(curr, index + 1, combos);**

**} else if (index == 1) {**

**curr[index] = 8;**

**generateDiceB(curr, index + 1, combos);**

**} else {**

**for (int i = 1; i <= 7; i++) {**

**final int finalI = i;**

**if (Arrays.stream(curr).noneMatch(x -> x == finalI))**

**}**

**}**

**}**

Step2: **Check Probabilities**

Here the probabilities of occurrences of each sum in the newly generated dice are compared with the probabilities of default dice.

**Step2: Check Probabilities**

Here the generated lists are traversed.

Input: List<int[]> diceacombos,List<int[]> dicebcombos, double[] frequency

For each pair of new dice arrays a[] and b[] a frequency dp1 array is created to store the

Frequency of occurrences of different sums.

**int[] dp1=new int[13];**

**int[] a=arr1;**

**int[] b=arr2;**

**int flag=0;**

**for(int n:a){**

**for(int m:b){**

**dp1[n+m]++; //frequency is updated**

**}**

**}**

**int[] dp1=new int[13];**

**int[] a=arr1;**

**int[] b=arr2;**

**for(int n:a){**

**for(int m:b){**

**dp1[n+m]++; //frequency is updated**

**}**

**}**

Now the frequency array from new dice a[] and b[] is compared with the original default dice frequency array dp. In original dice minimum sum=1 and maximum sum=12

**Step4:**

Loop from 2 to 12 and check if the frequency array of previous diceA and diceB is similar to new generated arrays.

If yes:

output the newly generated Arrays.

Else:

Exit

**for(int i=2;i<=12;i++){**

**if((int)dp[i]!=dp1[i]){**

**flag=1;**

**break;**

**}**

**}**

**if(flag==0){**

**printlist(a,b);**

**break;**

**}**

        }

**OPTIMIZATION:**

Initially, we had sums in frequency :

Sum 2=> 1

Sum 3 =>2

Sum 4=> 3

Sum 5 =>4

Sum 6=> 5

Sum 7=>6

Sum 8=>5

Sum 9=>4

Sum 10=>3

Sum 11=>2

Sum 12=>1

Sum 2 and Sum 12 occurs only once.

Minimum sum=2; Maximum sum=12;

Sum of 2 is possible only when (1,1) combination

Hence

New\_diceA[]={1,\_,\_,\_,\_,\_}

New\_diceB[]={1,\_,\_,\_,\_,\_}

Sum 12 is possible when (4,8),(3,9),(2,10),(1,11) combinations

If we pick (1,11)

New\_diceA[]={1,\_,\_,\_,\_,\_}

New\_diceB[]={1,11,\_,\_,\_,\_}

But diceA should have only one 1 since 12 is occurring only once . An increase in the instance of 1 will lead to (1,11),(1,11) multiple possibilities of 12.

Other spaces of diceA should be filled with {2,3,4}

But filling with anything greater than 1 will violate the sum combinations.

Will lead to combinations (2,11) = sum of 13; (3,11) =sum of 14; Maximum sum is 12.

If we pick(2,10)

New\_diceA={1,2,\_,\_,\_,\_}

New\_diceB={1,10,\_,\_,\_,\_}

Rest of the sides in dice A has to be filled with numbers. But filling it with another 1 or 2

Will violate first frequency rule.

New\_diceA = {1,2,1,\_,\_,\_}

New\_diceB = {1,10,\_,\_,\_,\_}

MinSum 2 has frequency 1. But adding 1 to dice A will lead to (1,1), (1,1)

New\_diceA= {1,2,2,\_,\_,\_}

New\_diceB= {1,10,\_,\_,\_,\_}

Adding another 2 will lead to 2 combinations of 12. (2,10) ,(2,10) .Frequency Violation

Anything greater than 2 cannot be added. As,

New\_diceA={1,2,3,\_,\_,\_}

New\_diceB={1,10,\_,\_,\_,\_}

Adding >2 Eg. 3 will lead to (3,10) =Sum 13. Combination is not permissible.

Similar frequency violations will occur for (3,9)

Hence (4,8) will lead to no frequency violation. As,

New\_diceA = {1, 4,\_,\_,\_,\_}

New\_diceB = {1, 8,\_,\_,\_,\_}

Now dice A has to be filled with {2 , 3} in 4 places with repetition.

Dice B has to be filled with {2 to 7} without repetition.

The brute force approach is used to try all possible combinations of dice arrays.

Dice A Combinations:

[1, 4, 2, 2, 2, 2]

[1, 4, 2, 2, 2, 3]

[1, 4, 2, 2, 3, 2]

[1, 4, 2, 2, 3, 3]

[1, 4, 2, 3, 2, 2]

[1, 4, 2, 3, 2, 3]

[1, 4, 2, 3, 3, 2]

[1, 4, 2, 3, 3, 3]

[1, 4, 3, 2, 2, 2]

[1, 4, 3, 2, 2, 3]

[1, 4, 3, 2, 3, 2]

[1, 4, 3, 2, 3, 3]

[1, 4, 3, 3, 2, 2]

[1, 4, 3, 3, 2, 3]

[1, 4, 3, 3, 3, 2]

[1, 4, 3, 3, 3, 3]

Dice B Combinations:

[1, 8, 2, 3, 4, 5]

[1, 8, 2, 3, 4, 6]

[1, 8, 2, 3, 4, 7]

[1, 8, 2, 3, 5, 4]

[1, 8, 2, 3, 5, 6]

[1, 8, 2, 3, 5, 7]

[1, 8, 2, 3, 6, 4]

[1, 8, 2, 3, 6, 5]

[1, 8, 2, 3, 6, 7]

[1, 8, 2, 4, 3, 5]

[1, 8, 2, 4, 3, 6]

[1, 8, 2, 4, 3, 7]

[1, 8, 2, 4, 5, 3]

[1, 8, 2, 4, 5, 6]

[1, 8, 2, 4, 5, 7]

[1, 8, 2, 4, 6, 3]

[1, 8, 2, 4, 6, 5]

[1, 8, 2, 4, 6, 7]

[1, 8, 2, 5, 3, 4]

[1, 8, 2, 5, 3, 6]

[1, 8, 2, 5, 3, 7]

[1, 8, 2, 5, 4, 3]

[1, 8, 2, 5, 4, 6]

[1, 8, 2, 5, 4, 7]

[1, 8, 2, 5, 6, 3]

[1, 8, 2, 5, 6, 4]

[1, 8, 2, 5, 6, 7]

[1, 8, 3, 2, 4, 5]

[1, 8, 3, 2, 4, 6]

[1, 8, 3, 2, 4, 7]

[1, 8, 3, 2, 5, 4]

[1, 8, 3, 2, 5, 6]

[1, 8, 3, 2, 5, 7]

[1, 8, 3, 2, 6, 4]

[1, 8, 3, 2, 6, 5]

[1, 8, 3, 2, 6, 7]

[1, 8, 3, 4, 2, 5]

[1, 8, 3, 4, 2, 6]

[1, 8, 3, 4, 2, 7]

[1, 8, 3, 4, 5, 2]

[1, 8, 3, 4, 5, 6]

[1, 8, 3, 4, 5, 7]

[1, 8, 3, 4, 6, 2]

[1, 8, 3, 4, 6, 5]

[1, 8, 3, 4, 6, 7]

[1, 8, 3, 5, 2, 4]

[1, 8, 3, 5, 2, 6]

[1, 8, 3, 5, 2, 7]

[1, 8, 3, 5, 4, 2]

[1, 8, 3, 5, 4, 6]

[1, 8, 3, 5, 4, 7]

[1, 8, 3, 5, 6, 2]

[1, 8, 3, 5, 6, 4]

[1, 8, 3, 5, 6, 7]

[1, 8, 4, 2, 3, 5]

[1, 8, 4, 2, 3, 6]

[1, 8, 4, 2, 3, 7]

[1, 8, 4, 2, 5, 3]

[1, 8, 4, 2, 5, 6]

[1, 8, 4, 2, 5, 7]

[1, 8, 4, 2, 6, 3]

[1, 8, 4, 2, 6, 5]

[1, 8, 4, 2, 6, 7]

[1, 8, 4, 3, 2, 5]

[1, 8, 4, 3, 2, 6]

[1, 8, 4, 3, 2, 7]

[1, 8, 4, 3, 5, 2]

[1, 8, 4, 3, 5, 6]

[1, 8, 4, 3, 5, 7]

[1, 8, 4, 3, 6, 2]

[1, 8, 4, 3, 6, 5]

[1, 8, 4, 3, 6, 7]

[1, 8, 4, 5, 2, 3]

[1, 8, 4, 5, 2, 6]

[1, 8, 4, 5, 2, 7]

[1, 8, 4, 5, 3, 2]

[1, 8, 4, 5, 3, 6]

[1, 8, 4, 5, 3, 7]

[1, 8, 4, 5, 6, 2]

[1, 8, 4, 5, 6, 3]

[1, 8, 4, 5, 6, 7]

The new dice a sides are: [1 4 2 2 3 3 ]

The new dice b sides are: [1 8 3 4 5 6 ]

The new dice a sides are: [1 4 2 3 2 3 ]

The new dice b sides are: [1 8 3 4 5 6 ]

The new dice a sides are: [1 4 2 3 3 2 ]

The new dice b sides are: [1 8 3 4 5 6 ]

The new dice a sides are: [1 4 3 2 2 3 ]

The new dice b sides are: [1 8 3 4 5 6 ]

The new dice a sides are: [1 4 3 2 3 2 ]

The new dice b sides are: [1 8 3 4 5 6 ]

The new dice a sides are: [1 4 3 3 2 2 ]

The new dice b sides are: [1 8 3 4 5 6 ]

THANK YOU