1.

* 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.

Input: ● Die\_A = [1, 2, 3, 4, 5, 6] & Die B = Die\_A = [1, 2, 3, 4, 5, 6] Output: ● A Transform Function undoom\_dice that takes (Die\_A, Die\_B) as input & outputs New\_Die\_A = [?, ?, ?, ?, ?, ?],New\_Die\_B = [?, ?, ?, ?, ?, ?] where, ● No New\_Die A[x] >4

Explanation:

This code introduces randomness to the values of two dice by generating new values based on probability calculations and a pseudorandom number generator. The goal is to simulate a transformation of the dice while maintaining some connection to the original probabilities.

1]Import Statements:

import java.util.ArrayList;: This imports the ArrayList class, though it is not used in the code.

import java.util.List;: This imports the List interface, though it is not used in the code.

2]UndoomDice Class:

The UndoomDice class is declared as a public class.

3]main Method:

The main method is the entry point of the program.

The initial die faces array is defined (initialDieFaces), and two copies (dieA and dieB) are created using the clone method.

4]undoomDice Method:

The undoomDice method takes two arrays (dieA and dieB) as input and returns a 2D array containing the new die configurations for dieA and dieB.

It calculates the total sum of values in dieA and dieB.

It computes the adjustment factor (factor) based on the ratio of total sums.

Two arrays (a and b) are created to store the adjusted values.

Values in array a are adjusted based on the minimum of 4 and the original values in dieA.

Values in array b are adjusted based on the factor and limited to a maximum of 6.

The new die configurations are returned as a 2D array.

5]Printing Results:

The main method prints the new die configurations for dieA and dieB using a simple loop.

Code and output;

