1. **Calculate and display the distribution of all possible combinations that can be obtained when rolling both Die A and Die B together.**

**Explanation:**

NumberOfFaces=6

TotalCombinations=36

Firstly I have defined a list of faces containing the possible outcomes of a six-sided die. Then I calculated and printed the total number of combinations when rolling two six-sided dice. Since each die has 6 faces, there are 6×6=36 possible combinations. The print statement introduces the next output section. Next, I initialized an empty dictionary sum\_count to keep track of the count of each possible sum of two dice and an empty list of combinations to store all possible combinations of two dice rolls. I have used nested loops to iterate over all possible outcomes of two dice rolls in which the outer loop represents the result of the first die, and the inner loop represents the result of the second die. For each combination, I have created a list containing the outcomes of both dice and append it to the combinations list. Updated the sum\_count dictionary to keep track of the occurrences of each possible sum. If the sum is not already a key in the dictionary, add it with a count of 1. If it's already a key, increment the count. The final loop prints the combinations in a structured way. It uses the range function to iterate over indices in steps of the length of faces. It then prints slices of the combinations list, effectively grouping them by the result of the first die. This gives you a structured output showing all combinations for each value of the first die.

**Code:**

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

total = len(faces) \*\* 2

print("\nCombinations Distribution:")

sum\_count={}

combinations = []

for i in range(1,7):

for j in range(1,7):

combinations.append(["Die A:" + str(i) ,"Die B:"+str(j)])

if i+j not in sum\_count:

sum\_count[i+j]=1

else:

sum\_count[i+j]+=1

for i in range(0, total, len(faces)):

print(combinations[i:i + len(faces)])

**Output:**

