**SECURIN ASSESSMENT**

**Github Link:**

https://github.com/abinayass/Assessment

**Collab Link(Code):** https://colab.research.google.com/drive/1eINf02FARWmX2P-7a8z5Q44yx\_mmrDii?usp=sharing

**PART – A**

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

1. Logic to the Given Statement:
   * + When we roll two dice, The total number of combinations can be obtained by multiplying the number of possible outcomes i.e., 1,2,3,4,5,6, of each dice.
     + Therefore the total combinations would be 6×6=36.
2. Explain how did you come up with the solution:

* Calculated the total combinations by using the multiplication operation of the number of faces (6) on each dice

**OUTPUT: 36**

**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**

1. Logic to the Given Statement:

* The distribution of all possible combinations is represented as a 6x6 matrix
* Each cell is a combination obtained by rolling Die A and Die B together

1. Explain how did you come up with the solution:

* Initialized an empty NumPy array with shape (6, 6) to represent the distribution matrix.
* Using dtype=object allows each element of the array to hold arbitrary Python objects, rather than being restricted to a single data type.
* Iterated over each possible value of Die A and Die B and print the all the possible combinations from (1,1),(1,2),(1,3) to (6,6)

**OUTPUT:**

**Distribution of all possible combinations:**

[[(1, 1) (1, 2) (1, 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)]]

**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**

1. Logic to the Given Statement:

* Since it is asked to calculate the probability of each possible sum occurring, the sum of the values on the two dice (i and j) is calculated and stored
* Then, Count the number of occurrences of each sum.
* And divide the count of occurrences by the total number of combinations to obtain the probability of each sum

1. Explain how did you come up with the solution:

* Calculate the possible sum of the values on the two dice when they rolled together
* Counted the occurrences of that sum in the distribution matrix.
* Then, divided the count by the total number of combinations to obtain the probability of that sum occurring.
* Finally, printed the probability of each sum with limited decimal places.

**FORMULA:**

**Probability= Number of occurrences of the sum/ Total number of combinations**

Example:

P(Sum = 2) = 1 / 36 or 0.0278

**PART – B**

**Now comes the real challenge. You were happily spending a lazy afternoon playing**

**your board game with your dice when suddenly the mischievous Norse God Loki ( You**

**love Thor too much & Loki didn’t like that much ) appeared.**

**Loki dooms your dice for his fun removing all the “Spots” off the dice.**

**No problem! You have the tools to re-attach the “Spots” back on the Dice.**

**However, Loki has doomed your dice with the following conditions:**

**● 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.**

**But in order to play your game, the probability of obtaining the Sums must remain the**

**same!**

**So if you could only roll P(Sum = 2) = 1/X, the new dice must have the spots reattached**

**such that those probabilities are not changed.**

**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**

1. Logic to the Given Statement:

* The code aims to simulate the rolling of two dice, represented by Die\_A and Die\_B, and then undo the outcomes of the dice rolls to find the possible combinations of outcomes that would result in the same set of sums as the original rolls.
* Initially, It calculates the original probabilities of each possible sum of the two dice.
* Then, it defines functions diceA\_possibilities and diceB\_possibilities to recursively find all possible combinations of outcomes for Die\_A and Die\_B, respectively, while ensuring that each die is rolled no more than four times.
* The undoom\_dice function calls these functions to find all possible combinations of outcomes for Die\_A and Die\_B that result in the same set of sums as the original rolls.
* Finally, it prints the modified dice outcomes and the corresponding probabilities after undoing the original dice rolls.

Explain how did you come up with the solution:

* The solution starts by calculating the original probabilities of each possible sum of the two dice.
* Then, it defines recursive functions diceA\_possibilities and diceB\_possibilities to find all possible combinations of outcomes for Die\_A and Die\_B, respectively.
* These functions iterate through all possible outcomes of the dice while ensuring that each die is rolled no more than four times (to match the original rolls).
* The undoom\_dice function calls these functions to find all possible combinations of outcomes for Die\_A and Die\_B that result in the same set of sums as the original rolls. It then compares these combinations with the original sums to find matches.
* Finally, it prints the modified dice outcomes and the corresponding probabilities based on the matches found.