**Part-A (15-20 Minutes):**

1. How many total combinations are possible?

* 36 combinations are possible when throwing two dice.

2. Calculate and display the distribution of all possible combinations that can be

obtained when rolling both Die A and Die B together.

3. Calculate the Probability of all Possible Sums occurring among the number of

combinations from (2).

**Code :**

**die\_A = [1, 2, 3, 4, 5, 6]**

**die\_B = [1, 2, 3, 4, 5, 6]**

**print("All possible combinations : ")**

**combination\_sum\_array = []**

**for i in range(len(die\_A)):**

**for j in range(len(die\_B)):**

**sum\_of\_pairs = die\_A[i] + die\_B[j]**

**if(sum\_of\_pairs not in combination\_sum\_array):**

**combination\_sum\_array.append(sum\_of\_pairs)**

**print((die\_A[i], die\_B[j]), end=" ")**

**print()**

**start = combination\_sum\_array[0]**

**end = combination\_sum\_array[-1]**

**while(start <= end):**

**print("Sum = ", start)**

**print("Combinations : ")**

**count = 0**

**for i in range(len(die\_A)):**

**for j in range(len(die\_B)):**

**sum\_of\_pair = die\_A[i] + die\_B[j]**

**if(sum\_of\_pair == start):**

**print((die\_A[i], die\_B[j]), end=" ")**

**count += 1**

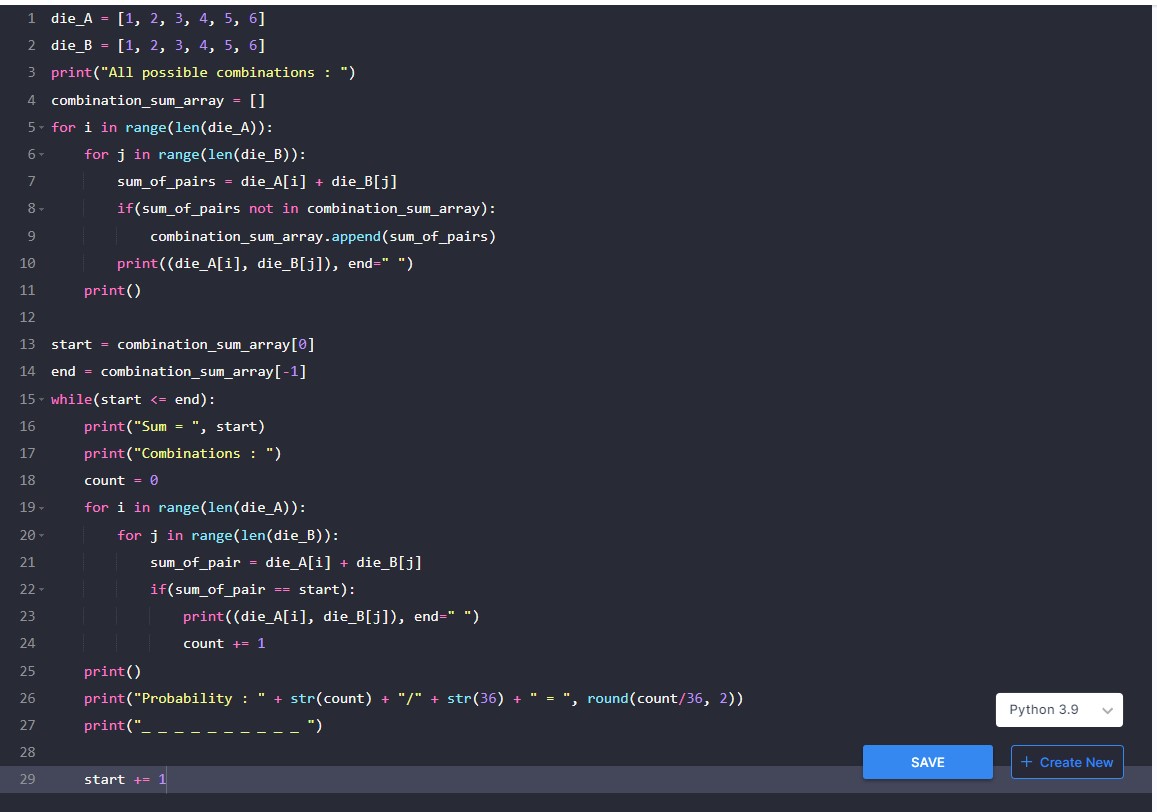
**print()**

**print("Probability : " + str(count) + "/" + str(36) + " = ", round(count/36, 2))**

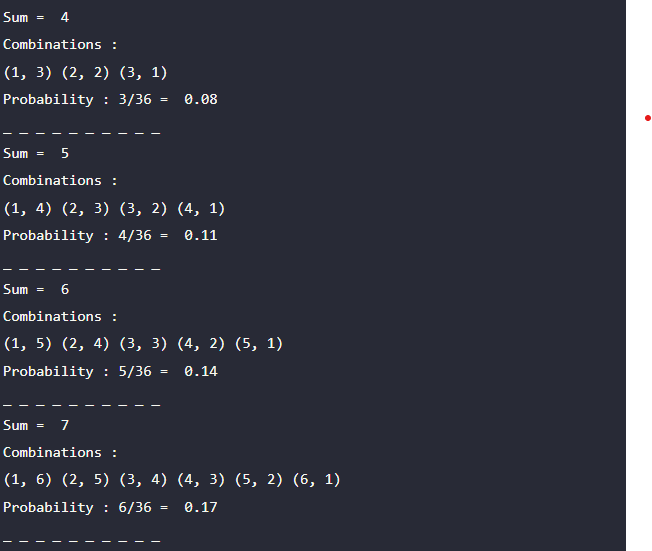
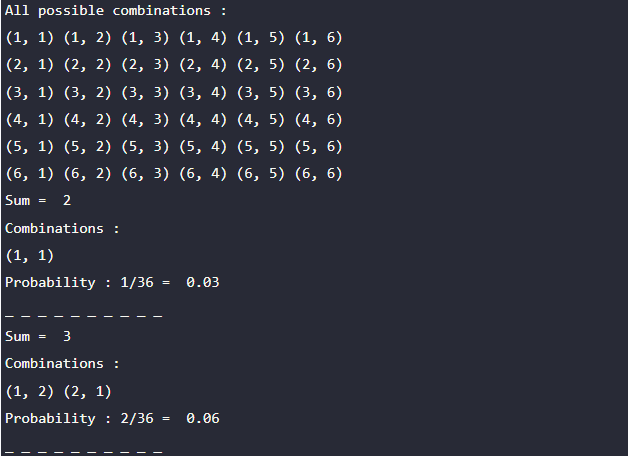
**print("\_ \_ \_ \_ \_ \_ \_ \_ \_ \_ ")**

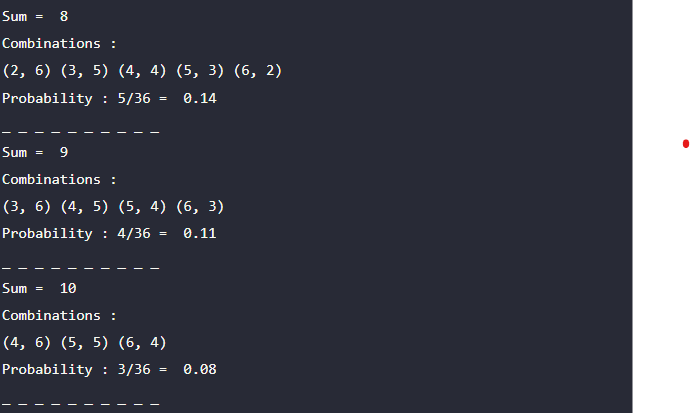
**start += 1**

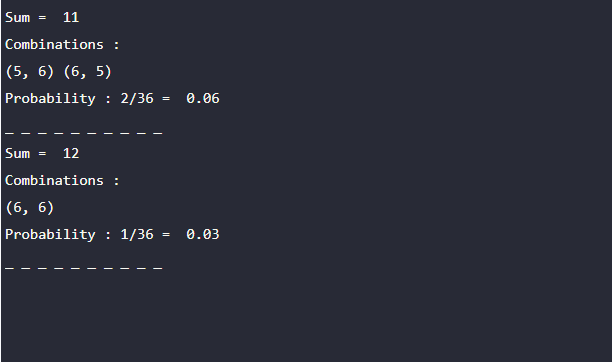
**Screenshot of code:**

****

**Output Screenshot:**

****

****

****

**Explanation:**

* Declare die\_A and die\_B as list and drop both the value [1, 2, 3, 4, 5, 6].
* Declare empty list to save sum of all combinations (combination\_sum\_array= []).
* Find all combinations, I took two for loop.

for i in range length of the die\_A list and for j in range length of the die\_B list.

and added a conditional statement to check whether sum\_of\_pair is not in **combination\_sum\_array** . If condition is True, allowed within the conditional statement.

* Print all combinations when throwing two dice.
* Now, Value of combination\_sum\_array = [] is [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12].
* Assigned first index value to **Start**, last index value to **end**. i.e, (Start = 2, end = 12).
* I have Used while loop range from (2 to 12).
* Print Sum of Combinations, Possible combinations and Probability of all Possible Sums.

**Example** :

Start = 2, end = 12

count = 0

While loop **(2 <= 12)** is True. Print Sum = 2 inside

Enter into the for loop, here calculate the sum\_of\_pair (1, 1) is the only possible pair of Sum = 2.

Assuming sum\_of\_pair == start, i.e., (2 == 2) allowed in conditional statement, and print possible pairs and count value increases by 1, now count = 1.

Print the Probability count value divided by 36 and round it up to two decimal places. **(1 / 36 ) => 0.03.**

Start value increases by 1.

**Output**:

Sum = **2**

Combinations :

**(1, 1)**

Probability **: 1/36 = 0.03**

\_ \_ \_ \_ \_ \_ \_ \_ \_ \_

**Part – B:**

**Code :**

**def undoom\_dice(Die\_A, Die\_B):**

**new\_die\_A = []**

**new\_die\_B = []**

**combination\_sum\_array = []**

**for i in range(len(Die\_A)):**

**for j in range(len(Die\_B)):**

**sum\_of\_pairs = Die\_A[i] + Die\_B[j]**

**if(sum\_of\_pairs not in combination\_sum\_array):**

**combination\_sum\_array.append(sum\_of\_pairs)**

**for i in Die\_A:**

**for j in Die\_B:**

**sum\_of\_value = i + j**

**if(sum\_of\_value in combination\_sum\_array):**

**if i > 4:**

**new\_die\_A.append(i - 4)**

**break**

**else:**

**if(i not in new\_die\_A):**

**new\_die\_A.append(i)**

**if(j not in new\_die\_B):**

**new\_die\_B.append(j)**

**return new\_die\_A, new\_die\_B**

**Die\_A = [1, 2, 3, 4, 5, 6]**

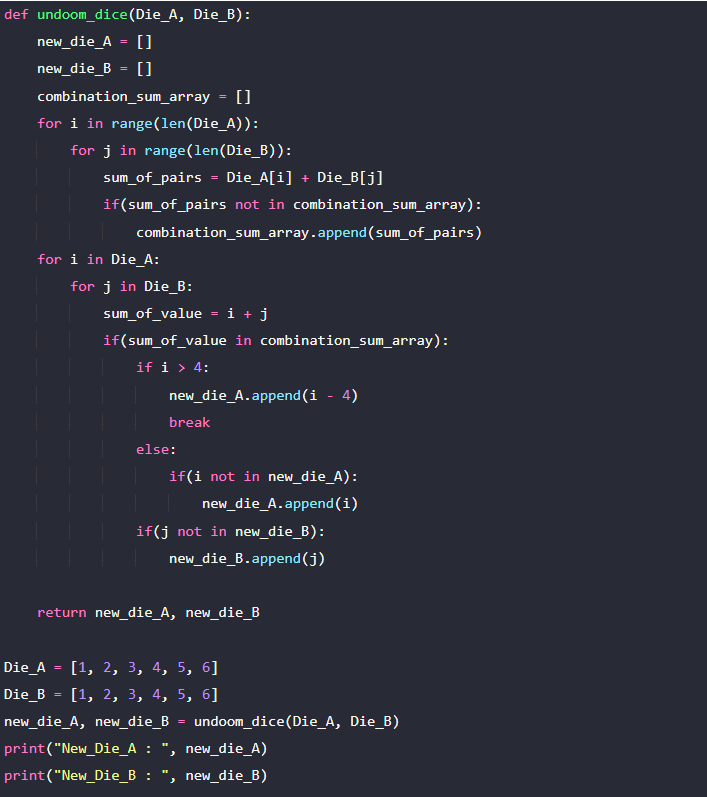
**Die\_B = [1, 2, 3, 4, 5, 6]**

**new\_die\_A, new\_die\_B = undoom\_dice(Die\_A, Die\_B)**

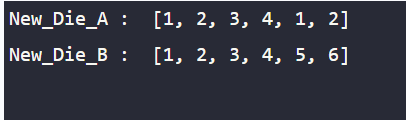
**print("New\_Die\_A : ", new\_die\_A)**

**print("New\_Die\_B : ", new\_die\_B)**

**Code Screenshot :**

****

**Output:**

****

**Explanation:**

* Define a function undoom\_dice and pass it as arguments Die\_A, Die\_B with an initial value.
* Declare new-die\_A and new\_Die\_B with an empty list. Similarly, combination\_sum\_array.
* for i in range length of the Die\_A list and for j in range length of the Die\_B list.

and added a conditional statement to check whether sum\_of\_pair is not in combination\_sum\_array . If condition is True, allowed within the conditional statement.

* Two for loops were used, one for Die\_A and one for Die\_B.
* Once the sum\_of\_value is calculated, I added an If condition that the sum\_of\_value is present in the combination\_sum\_array allowed inside the statement, Die\_A value > 4 is subtracted from 4 and added to the new\_die\_A.
* If not, block added a value to new\_die\_A that did not exist previously.
* The same process was followed for new\_die\_B.
* As a final step, return the new\_die\_A, new\_die\_B from the function. After that, print the new\_die\_A and new\_die\_B.