PART-A

Total number of combinations

```
def total_combinations():
  """Calculates the total number of combinations for rolling two six-sided dice."""
  num_sides = 6
  return num_sides * num_sides


total_combinations = total_combinations()
print("Total combinations:", total_combinations)  # Output: Total combinations: 36
```

DISTRIBUTION OF COMBINATION

```
num_faces_die_a = 6
num_faces_die_b = 6
distribution_matrix = [[0] * num_faces_die_b for _ in range(num_faces_die_a)]


for i in range(num_faces_die_a):
    for j in range(num_faces_die_b):
        distribution_matrix[i][j] = (i + 1) + (j + 1)


print("Distribution Matrix:")
for row in distribution_matrix:
    print(" ".join(map(str, row)))
```

PROBABILITY OF SUMS:

```
def probability_of_sums(sum):
```

```
  """Calculates the probability of a specific sum occurring when rolling two six-sided dice."""
  num_sides = 6
  total_combinations = num_sides * num_sides
  if sum < 2 or sum > 12:
    return 0  # Handle invalid sums (outside range)
  successful_combinations = 0
  for die_a in range(1, num_sides + 1):
    for die_b in range(1, num_sides + 1):
      if die_a + die_b == sum:
        successful_combinations += 1
  return successful_combinations / total_combinations

for sum in range(2, 13):  # Calculate probability for all valid sums
  probability = probability_of_sums(sum)
  print(f"P(Sum = {sum}) = {probability}")
```

OUTPUT:

Total combinations: 36

Distribution Matrix:

2 3 4 5 6 7

3 4 5 6 7 8

4 5 6 7 8 9

5 6 7 8 9 10

6 7 8 9 10 11

7 8 9 10 11 12

P(Sum = 2) = 0.027777777777777776

P(Sum = 3) = 0.05555555555555555

P(Sum = 4) = 0.08333333333333333

P(Sum = 5) = 0.1111111111111111

P(Sum = 6) = 0.1388888888888889

P(Sum = 7) = 0.16666666666666666

P(Sum = 8) = 0.1388888888888889

P(Sum = 9) = 0.1111111111111111

P(Sum = 10) = 0.08333333333333333

P(Sum = 11) = 0.05555555555555555

P(Sum = 12) = 0.027777777777777776