1) write a python program to find the mean median and mode of the given number def calculate\_3Ms(array): # Calculate Mean mean = statistics.mean(array) # Calculate Median median = statistics.median(array)

# Calculate Mode mode = statistics.mode(array) return mean, median, mode

# Given array array = [16, 18, 27, 23, 21, 19]

# Calculate 3Ms mean, median, mode = calculate\_3Ms(array)

# Print results

Output: mean=20, median=19, mode=16

2 write a python program to find all the the combination of digits of a given number.

def find\_combinations(num):

# Convert number to string to access individual digits digits = str(num)

# Generate all permutations permutations = [''.join(p) for p in itertools.permutations(digits)] return permutations num = 123 combinations = find\_combinations(num) print("Number:", num) print("Combinations:") for combination in combinations:

print(combination)

output: Number: 123 Combinations:

123

132

213

231

312

321

3) write a program to generate multiplication table

# Multiplication Table Generator

# Function to generate multiplication table def generate\_multiplication\_table(number, up\_to=10):

print(f"Multiplication Table for {number}") for i in range(1, up\_to + 1):

result = number \* i print(f"{number} x {i} = {result}")

# User input for number and range try:

number = int(input("Enter the number for multiplication table: ")) up\_to = int(input("Enter the range (e.g., 10 for 1 to 10): ")) generate\_multiplication\_table(number, up\_to) except ValueError:

print("Please enter valid numbers.")

4) write python program to find L.C.M and G.C.M of three numbers import math

# Function to find GCD of three numbers def find\_gcd(a, b, c):

return math.gcd(math.gcd(a, b), c)

# Function to find LCM of two numbers def find\_lcm(a, b):

return abs(a \* b) // math.gcd(a, b)

# Function to find LCM of three numbers def find\_lcm\_of\_three(a, b, c): return find\_lcm(find\_lcm(a, b), c)

# User input for three numbers try:

num1 = int(input("Enter the first number: ")) num2 = int(input("Enter the second number: ")) num3 = int(input("Enter the third number: "))

# Calculating GCD and LCM gcd\_result = find\_gcd(num1, num2, num3) lcm\_result = find\_lcm\_of\_three(num1, num2, num3)

print(f"The GCD of {num1}, {num2}, and {num3} is: {gcd\_result}") print(f"The LCM of {num1}, {num2}, and {num3} is: {lcm\_result}")

except ValueError:

print("Please enter valid integer numbers.")

5 write python program to read the numbers until -1 is encountered find the average of positive numbers and negative number entered by user # Function to calculate the average of a list of numbers def calculate\_average(numbers): if len(numbers) == 0:

return 0 # Avoid division by zero return sum(numbers) / len(numbers)

# Lists to store positive and negative numbers positive\_numbers = [] negative\_numbers = []

print("Enter numbers one by one. Enter -1 to stop:")

# Read numbers from the user until -1 is entered while True: try:

num = float(input("Enter a number: ")) if num == -1:

break # Stop the input loop elif num > 0: positive\_numbers.append(num) elif num < 0:

negative\_numbers.append(num) except ValueError:

print("Please enter a valid number.")

# Calculate the averages

positive\_avg = calculate\_average(positive\_numbers) negative\_avg = calculate\_average(negative\_numbers)

# Display the results print(f"Average of positive numbers: {positive\_avg}") print(f"Average of negative numbers: {negative\_avg}")

6) give an array of integers nums containing n+1 integers where each integers is in the range [1,n] inclusive there is only one repeated numbers in nums return this repeated number

def find\_duplicate(nums):

# Initialize the tortoise and hare pointers tortoise = nums[0] hare = nums[0]

# Phase 1: Finding the intersection point in the cycle while True:

tortoise = nums[tortoise] hare = nums[nums[hare]] if tortoise == hare:

break

# Phase 2: Find the entrance to the cycle (duplicate number) tortoise = nums[0] while tortoise != hare: tortoise = nums[tortoise] hare = nums[hare]

return hare

7) write a python program to print matrix in spiral form

def spiral\_print(matrix):

if not matrix: return

top, bottom = 0, len(matrix) - 1 left, right = 0, len(matrix[0]) - 1

while top <= bottom and left <= right:

# Print the top row for i in range(left, right + 1): print(matrix[top][i], end=" ") top += 1

# Print the right column for i in range(top, bottom + 1): print(matrix[i][right], end=" ") right -= 1

# Print the bottom row (if still within bounds) if top <= bottom: for i in range(right, left - 1, -1): print(matrix[bottom][i], end=" ") bottom -= 1

# Print the left column (if still within bounds) if left <= right: for i in range(bottom, top - 1, -1): print(matrix[i][left], end=" ") left += 1

# Example usage matrix = [

[1, 2, 3, 4],

[5, 6, 7, 8],

[9, 10, 11, 12],

[13, 14, 15, 16]

]

spiral\_print(matrix)