Q) 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

Q) 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

Q) 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.")
```

Q) 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.")
```

Q) 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}")
```

Q) 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
```

Q) 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)
```