Python Programming

Day-4

1) PERFECT NUMBER -

```
def is_perfect_number(n):
    if n < 1:
        return False

    divisors_sum = 0
    for i in range(1, n):
        if n % i == 0:
            divisors_sum += i

    return divisors_sum == n

number = int(input("Enter a number: "))
if is_perfect_number(number):
    print(f"{number} is a perfect number.")
else:
    print(f"{number} is not a perfect number.")
```

2) TRANSPOSAL MATRIX-

```
def print_matrix(matrix):
    for row in matrix:
        print(" ".join(map(str, row)))

def transpose_matrix(matrix):
    rows = len(matrix)
    cols = len(matrix[0])
    transposed = [[0 for _ in range(rows)] for _ in range(cols)]

    for i in range(rows):
        for j in range(cols):
            transposed[j][i] = matrix[i][j]

    return transposed

    matrix = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
```

```
]

print("Original Matrix:")
print_matrix(matrix)

transposed_matrix = transpose_matrix(matrix)

print("\nTransposed Matrix:")
print_matrix(transposed_matrix)
```

3) ROW AND COLUMN -

```
def print_matrix(matrix):
    for row in matrix:
        print(" ".join(map(str, row)))

def row_sums(matrix):
    return [sum(row) for row in matrix]

def column_sums(matrix):
```

```
    num_cols = len(matrix[0])

    return [sum(matrix[row][col] for row in range(len(matrix))) for col in
range(num_cols)]


def diagonal_sums(matrix):

    primary_diagonal = sum(matrix[i][i] for i in range(len(matrix)))

    secondary_diagonal = sum(matrix[i][len(matrix) - i - 1] for i in
range(len(matrix)))

    return primary_diagonal, secondary_diagonal


matrix = [

    [1, 2, 3],

    [4, 5, 6],

    [7, 8, 9]

]


print("Matrix:")

print_matrix(matrix)


row_sums_result = row_sums(matrix)

column_sums_result = column_sums(matrix)

primary_diagonal_sum, secondary_diagonal_sum = diagonal_sums(matrix)
```

```
print("\nRow sums:", row_sums_result)
print("Column sums:", column_sums_result)
print("Primary diagonal sum:", primary_diagonal_sum)
print("Secondary diagonal sum:", secondary_diagonal_sum)
```

4) BOUNDARY ELEMENT OF THE MATRIX -

```
def print_matrix(matrix):
    for row in matrix:
        print(" ".join(map(str, row)))

def boundary_sum(matrix):
    if not matrix or not matrix[0]:
        return 0

    rows = len(matrix)
    cols = len(matrix[0])
    total_sum = 0
```

```
    for col in range(cols):
        total_sum += matrix[0][col]



    if rows > 1:
        for col in range(cols):
            total_sum += matrix[rows - 1][col]

    for row in range(1, rows - 1):
        total_sum += matrix[row][0]  # Left column
        if cols > 1:
            total_sum += matrix[row][cols - 1]  # Right column

    return total_sum

matrix = [
    [1, 2, 3],
    [4, 5, 6],
    [7, 8, 9]
]
```

```
print("Matrix:")
print_matrix(matrix)

boundary_sum_result = boundary_sum(matrix)

print("\nSum of boundary elements:", boundary_sum_result)
```

5. SPIRAL ORDER -

```
def spiral_order(matrix):
    if not matrix:
        return []

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

    while top <= bottom and left <= right:
   for i in range(left, right + 1):
            result.append(matrix[top][i])
```

```
        top += 1

        for i in range(top, bottom + 1):
            result.append(matrix[i][right])
        right -= 1

        if top <= bottom:

            for i in range(right, left - 1, -1):
                result.append(matrix[bottom][i])
            bottom -= 1

        if left <= right:

            for i in range(bottom, top - 1, -1):
                result.append(matrix[i][left])
            left += 1

    return result

matrix = [
```

```
    [1, 2, 3, 4],
    [5, 6, 7, 8],
    [9, 10, 11, 12],
    [13, 14, 15, 16]
]

print("Matrix:")
for row in matrix:
    print(" ".join(map(str, row)))

spiral_order_result = spiral_order(matrix)

print("\nMatrix in spiral order:")
print(" ".join(map(str, spiral_order_result)))
```

6. SUM OF THE N NUMBERS -

```
def sum_of_n_numbers():
    n = int(input("Enter the number of elements: "))

    total_sum = 0
```

```
    for i in range(n):
        num = float(input(f"Enter number {i+1}: "))
        total_sum += num

    print(f"The sum of the {n} numbers is: {total_sum}")

sum_of_n_numbers()
```

7. SUM OF THR FACTORAL-

```
def factorial(num):
    """Function to calculate the factorial of a number."""
    if num == 0 or num == 1:
        return 1
    else:
        return num * factorial(num - 1)

def sum_of_factorials(n):
    """Function to calculate the sum of factorials from 1! to n!."""
    total_sum = 0
    for i in range(1, n + 1):
```

```
        total_sum += factorial(i)
    return total_sum

n = int(input("Enter a number: "))
result = sum_of_factorials(n)
print(f"The sum of factorials from 1! to {n}! is: {result}")
```

8. SUM OF THE SQUARE ROOTS-

```
def sum_of_squares(n):
    """Function to calculate the sum of squares from 1^2 to n^2."""
    total_sum = sum(i**2 for i in range(1, n + 1))
    return total_sum

n = int(input("Enter a number: "))
result = sum_of_squares(n)
print(f"The sum of squares from 1^2 to {n}^2 is: {result}")
```

9) *MEAN MEDIUM AND MODE OF THE ELEMENT -*

```
from statistics import mean, median, mode, StatisticsError
```

```
def calculate_statistics(numbers):
    """Function to calculate mean, median, and mode of a list of numbers."""
    try:
        mean_value = mean(numbers)
    except StatisticsError:
        mean_value = None

    try:
        median_value = median(numbers)
    except StatisticsError:
        median_value = None

    try:
        mode_value = mode(numbers)
    except StatisticsError:
        mode_value = None

    return mean_value, median_value, mode_value

  numbers = [1, 2, 2, 3, 4, 4, 4, 5, 6]
```

```
mean_value, median_value, mode_value = calculate_statistics(numbers)

print(f"Mean: {mean_value}")
print(f"Median: {median_value}")
print(f"Mode: {mode_value}")
```

10) nth LARGEST NUMBER-

```
def nth_largest(numbers, n):
    """Function to find the nth largest number in a list."""
    if n > len(numbers):
        return None
    sorted_numbers = sorted(numbers, reverse=True)
    return sorted_numbers[n - 1]

numbers = [4, 2, 5, 1, 3, 7, 6]
n = int(input("Enter the value of n: "))

result = nth_largest(numbers, n)
```

```
if result is not None:
    print(f"The {n}th largest number is: {result}")
else:
    print(f"The list does not have {n} elements.")
```