Assignment

Challenge: Spiral Order Traversal of a Matrix Using Recursion (10 Points)

Problem:
Write a recursive Python function to return the elements of a **2D matrix** in **spiral order**.

Rules:

1. The function should recursively reduce the matrix by removing the outermost rows and columns, collecting their elements in spiral order.
2. The function must handle matrices of varying sizes, including edge cases like empty matrices or single-row/column matrices.

Steps of Spiral Order Traversal

For a given matrix, you follow these steps to traverse it in spiral order:

1. **Traverse the Top Row**: Start from the first column of the first row and move right.
2. **Traverse the Right Column**: Move downward along the last column.
3. **Traverse the Bottom Row**: If there's still a row left, traverse it from right to left.
4. **Traverse the Left Column**: If there's still a column left, move upward along the first column.

After completing the outermost layer, recursively repeat the same steps for the inner layers of the matrix.

Example Walkthrough

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

Traversal Process:

1. **Top Row**: [1, 2, 3]
   Collect elements from the first row: **1, 2, 3**.
2. **Right Column**: [6, 9]
   Move downward along the last column: **6, 9**.

3. **Bottom Row**: [8, 7]
   Move from right to left along the last row: **8, 7**.
4. **Left Column**: [4]
   Move upward along the first column: **4**.
5. **Inner Matrix**: The remaining inner matrix is: [5]

Edge Cases:

1. **Single Row Matrix**:

```
matrix = [[1, 2, 3, 4]]
print(spiral_order(matrix))  # Output: [1, 2, 3, 4]
```

2. **Single Row Matrix**:

```
matrix = [[1], [2], [3], [4]]
print(spiral_order(matrix))  # Output: [1, 2, 3, 4]
```

3. **Empty Matrix**:

```
matrix = []
print(spiral_order(matrix))  # Output: []
```

4. **Non-Square Matrix**:

```
matrix = [
    [1, 2, 3],
    [4, 5, 6]
]
print(spiral_order(matrix))  # Output: [1, 2, 3, 6, 5, 4]
```