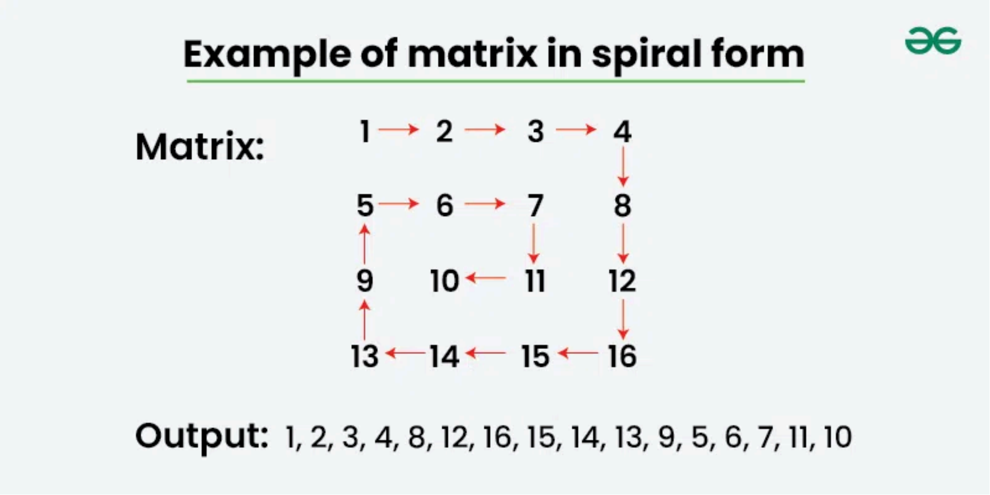
🔄 What is the Spiral Matrix Problem?

The **Spiral Matrix problem** means **printing or traversing a 2D matrix in spiral order**.

👉 Spiral order means:

- Start from **top-left**
- Go **right**
- Then **down**
- Then **left**
- Then **up**
- Repeat this process until all elements are covered



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

Output: 1, 2, 3, 4, 8, 12, 16, 15, 14, 13, 9, 5, 6, 7, 11, 10

✅ Example

Input matrix:

```
1  2  3
4  5  6
7  8  9
```

Output (Spiral Order):

```
1 2 3 6 9 8 7 4 5
```

🧠 Logic (Simple Understanding)

We maintain **4 boundaries**:

- `top` → first row
- `bottom` → last row
- `left` → first column
- `right` → last column

And we move in this order:

1. Left → Right (top row)
2. Top → Bottom (right column)
3. Right → Left (bottom row)
4. Bottom → Top (left column)

After each step, we **shrink the boundary**.

✅ Java Code (Simple & Clean)

```
public class SpiralMatrix {

    public static void printSpiral(int[][] matrix) {

        int top = 0;
        int bottom = matrix.length - 1;
        int left = 0;
        int right = matrix[0].length - 1;

        while (top <= bottom && left <= right) {

            // left to right
            for (int i = left; i <= right; i++) {
                System.out.print(matrix[top][i] + " ");
            }
            top++;

            // top to bottom
            for (int i = top; i <= bottom; i++) {
                System.out.print(matrix[i][right] + " ");
            }
            right--;

            // right to left
            if (top <= bottom) {
                for (int i = right; i >= left; i--) {
                    System.out.print(matrix[bottom][i] + " ");
                }
                bottom--;
            }

            // bottom to top
            if (left <= right) {
                for (int i = bottom; i >= top; i--) {
                    System.out.print(matrix[i][left] + " ");
                }
                left++;
            }
        }
    }

    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3},
            {4, 5, 6},
            {7, 8, 9}
        };

        printSpiral(matrix);
    }
}
```

🖨️ Output

```
1 2 3 6 9 8 7 4 5
```

⏱️ Time & Space Complexity

- **Time:** O(n × m)
- **Space:** O(1) (no extra space used)