Question: https://leetcode.com/problems/spiral-matrix/

Maintain 4 pointers:

Top-0

Bottom-arr.length

Left-0

Right-arr[0].length

So in order to traverse spirally we cover either one row or one column at a time,

1.So what we do is when we complete traversing the top most row we increase top pointer.

And start traversing the right-1 column.

2.When we complete traversing the right-1 column we decrease right pointer.

And start traversing the bottom-1 row.

3.When we complete traversing the bottom-1 row we decrease bottom pointer.

And start traversing the left most row.

4.When we complete traversing the left column we increase left pointer.

And start traversing the top row.

We repeat this four steps till top<bottom and left<right.

Code:  
class Solution {

public List<Integer> spiralOrder(int[][] matrix) {

List<Integer> res = new ArrayList<Integer>();

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

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

for(int j=left; j<right; j++){

res.add(matrix[top][j]);

}

top++;

if(left>=right || top>=bottom) break;

for(int i=top; i<bottom; i++){

res.add(matrix[i][right-1]);

}

right--;

if(left>=right || top>=bottom) break;

for(int j=right-1; j>=left; j--){

res.add(matrix[bottom-1][j]);

}

bottom--;

if(left>=right || top>=bottom) break;

for(int i=bottom-1; i>=top; i--){

res.add(matrix[i][left]);

}

left++;

}

return res;

}

}

Github Link :<https://lnkd.in/ecwtJeaz>