**Shift 2D Grid**

Given a 2D grid of size m x n and an integer k. You need to shift the grid k times.

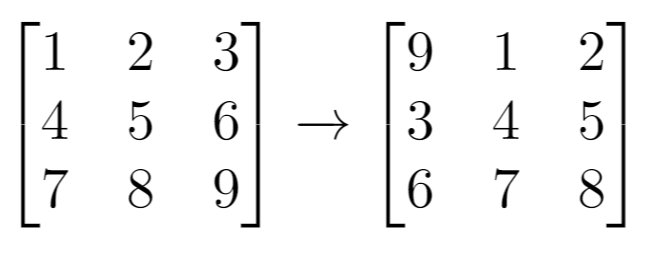
In one shift operation:  
Element at grid[i][j] moves to grid[i][j + 1].

Element at grid[i][n - 1] moves to grid[i + 1][0].

Element at grid[m - 1][n - 1] moves to grid[0][0].

Return the 2D grid after applying shift operation k times.

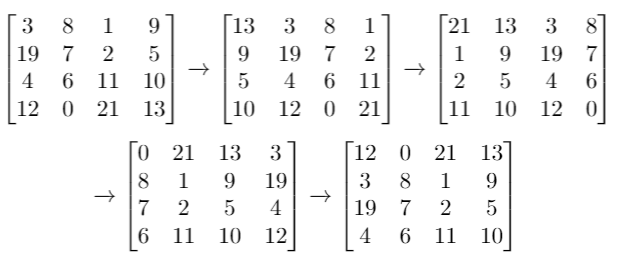
Example 1:



Input: grid = [[1,2,3],[4,5,6],[7,8,9]], k = 1

Output: [[9,1,2],[3,4,5],[6,7,8]]

Example 2:



Input: grid = [[3,8,1,9],[19,7,2,5],[4,6,11,10],[12,0,21,13]], k = 4

Output: [[12,0,21,13],[3,8,1,9],[19,7,2,5],[4,6,11,10]]

Example 3:  
Input: grid = [[1,2,3],[4,5,6],[7,8,9]], k = 9

Output: [[1,2,3],[4,5,6],[7,8,9]]

Constraints:  
m == grid.length

n == grid[i].length

1 <= m <= 50

1 <= n <= 50

-1000 <= grid[i][j] <= 1000

0 <= k <= 100

**Solution:**

// Array of Matrix:

// 9,1,2 9,1,2 8,9,1 7,8,9   
// 3,4,5 k=1--> 3,4,5 k=2--> 2,3,4 k=3--> 1,2,3   
// 6,7,8 6,7,8 5,6,7 4,5,6

function matPus(grid, k){

count =1;

while(count<=k){

grid[0].unshift(grid[grid.length-1][grid[grid.length-1].length-1]);

for(let i =1; i<grid.length; i++){

grid[i].unshift(grid[i-1][grid[i-1].length-1]);

}

for(let j=0; j<grid.length;j++){

grid[j].pop(grid[j][grid[j].length-1]);

}

count++;

}

return grid;

}

matPus([[1,2,3],[4,5,6],[7,8,9]], 3);  
matPus([[1],[2],[3],[4],[7],[6],[5]], 23);

**Output:**

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

[[6], [5], [1], [2], [3], [4], [7]]