# Matrix Spiral Print

Given a 2D array (matrix) named `M`, print all items of `M` in a spiral order, clockwise.

For example:

```
M  =  1   2   3   4   5
      6   7   8   9  10
     11  12  13  14  15
     16  17  18  19  20
```

The clockwise spiral print is: `1 2 3 4 5 10 15 20 19 18 17 16 11 6 7 8 9 14 13 12`

## Hints & Tips

- Most of the work on this question is manipulating the indices. Make sure that your peer is not using negative or out of bounds indices at any point.
- Make sure that all indices are printed and that no index is printed twice.
- If your peer is stuck, ask what patterns and directions repeat on the spiral printing. If that doesn't help, ask what shape does the spiral matrix have (rectangular) and how can you create it.
- Any solution that involves changing the matrix is definitely not advised. There is no need to do that. If you peer gets there, try to ask them why.

## Solution

Let `M` be a matrix of `m` rows and `n` columns.

The spiral print can be implemented by repeating a print of 4 edges, in a converging manner:

- Print the uppermost row from left to right
- Print the rightmost column from top to bottom
- Print the lowermost row from right to left
- Print the leftmost column from bottom to top

To direct the spiral order and figure what is the next row/column to print we maintain 4 indices:

`topRow` - index of the the upper most row to be printed, starting from `0` and incrementing
`btmRow` - index of the the lowermost row to be printed, stating from `m-1` and decrementing
`leftCol` - index of the leftmost column to be printed, starting from `0` and incrementing
`rightCol` - index of the the rightmost row to be printed, starting from `n-1` and decrementing

**Pseudo code:**

```
function spiralMatrixPrint(M):
   topRow = 0
   btmRow = m-1
   leftCol = 0
   rightCol = n-1

   while (topRow <= btmRow AND leftCol <= rightCol):
      # print the next top row
      for i from leftCol to rightCol:
         print M[topRow][i]
      topRow++
```

```
    # print the next right hand side column
    for i from topRow to btmRow:
       print M[i][rightCol]
    rightCol--

    # print the next bottom row
    if (topRow <= btmRow):
       for i from rightCol to leftCol:
          print M[btmRow][i]
       btmRow--

    # print the next left hand side column
    if (leftCol <= rightCol):
       for i from btmRow to topRow:
          print M[i][leftCol]
       leftCol++
```

**Runtime Complexity**: iterating over $n \cdot m$ cells and printing them takes **O(n · m)**.

**Space Complexity**: using a constant number of indices (4), therefore: **O(1)**.