Convert 1D array into 2D Array

Input

n → size of array
arr[] of size n
p & q input is the dimension of 2D array
p → no of rows
q → no of columns

n = 9 0 1 2 3 4 5 6 7 8
arr = [1, 2, 3, 4, 5, 6, 7, 8, 9]
p = 3 → row
q = 3 → col.

Output

mat[][] =

	0	1	2
0	1 00	2 01	3 02
1	4 10	5 11	6 12
2	7 20	8 21	9 22

0 → 00
1 → 01
2 → 02
3 → 10

(0 to 2) /3 → 0 row
(3 to 5) /3 → 1
(6 to 8) /3 → 2

arr[0] → mat[0][0]
arr[1] → mat[0][1]
arr[2] → mat[0][2]
arr[3] → mat[1][0]
arr[4] → mat[1][1]
arr[5] → mat[1][2]
arr[6] → mat[2][0]
arr[7] → mat[2][1]
arr[8] → mat[2][2]

4 → 11
5 → 12

row = i/3;

i = 0 to n-1

i = 0
- i = 0/3 = 0
- i = 1, 1/3 = 0
- i = 2, 2/3 = 0

} 1st row.

- i = 3, 3/3 = 1
- i = 4, 4/3 = 1
- i = 5, 5/3 = 1

} 2nd row

- i = 6, 6/3 = 2
- i = 7, 7/3 = 2
- i = 8, 8/3 = 2

} 3rd row.

Col.

- 0 → 0
- 1 → 1
- 2 → 2

} 0%3 = 0, 1%3 = 1, 2%3 = 2

- 3 → 0
- 4 → 1
- 5 → 2

} 3%3 = 0, 4%3 = 1, 5%3 = 2

- 6 → 0

} 6%3 = 0

6 → 0
7 → 1
8 → 2

6 % 3 = 0
7 % 3 = 1
8 % 3 = 2

mat

Array index

1 ← 0 → 0/3 = 0 [0][0] = 1
0 % 3 = 0

2 ← 1 → 1/3 = 0 [0][1] = 2
1 % 3 = 1

3 ← 2 → 2/3 = 0 [0][2] = 3
2 % 3 = 2

4 ← 3 → 3/3 = 1 [1][0] = 4
3 % 3 = 0

4 → 4/3 = 1 [1][1]
4 % 3 = 1

Code:

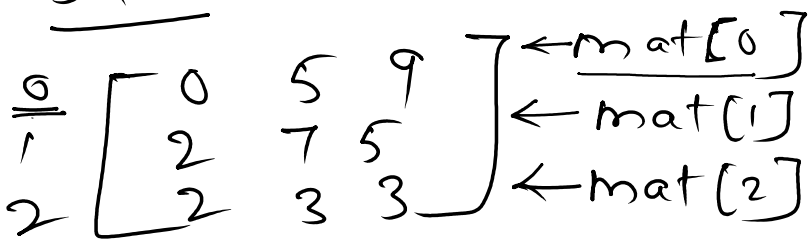
```
int mat[][] = new int[p][q];
for(int i=0; i<n; i++){
    mat[i/q][i%q] = arr[i];
```

```java
public class Solution {

    public static void main(String[] args) {
        /* Enter your code here. Read input from STDIN. F
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        int arr[] = new int[n];
        for(int i=0;i<n;i++){
            arr[i] = sc.nextInt();
        }
        int p = sc.nextInt();
        int q = sc.nextInt();
        int mat[][] = new int[p][q];
        for(int i=0;i<n;i++){
            mat[i/q][i%q]= arr[i];
        }
        for(int i=0;i<p;i++){
            for(int j=0;j<q;j++){
                System.out.print(mat[i][j]+" ");
            }
            System.out.println();
        }
    }
}
```

Shift Matrix Row-Wise

n = 3

3x3.




```
int mat[][] = new int[n][n];
for (int i=0; i<n; i++) {
    rotate(mat[i], k);
}
```

→ Rotate/Shift array elements by K

2 pointer Technique
- → reverse entire array
- → reverse (0 to k-1)
- → reverse (k to n-1)

Modify the Matrix

m = 3
n = 4
mat[3][4] =

	0	1	2	3
0	1	0	0	1
1	0	0	1	0
2	0	0	0	0

mat[0][0] = 1 , mat[0][3] = 1 , mat[1][2] = 1

row col

i = 0 to 3 , j = 0 to 2
(0 to col, 0 to row)

00	01	02	03
10	11	12	13
20	21	22	23

arr[j][i]

Output →

	0	1	2	3
0	1_{00}	1_{01}	1_{02}	1_{03}
1	1_{10}	1_{11}	1_{12}	1_{13}
2	1_{20}	0_{21}	1_{22}	1_{23}

Algorithm :- 1. Make an array of rows, row[] and another array of columns col[]

2. Traverse the matrix
if, at any index we get 1.
row[i] = 1 , col[0] = 1
row[0] = 1
row[0] = 1 , col[3] = 1
row[1] = 1 , col[2] = 1

	0	1	2

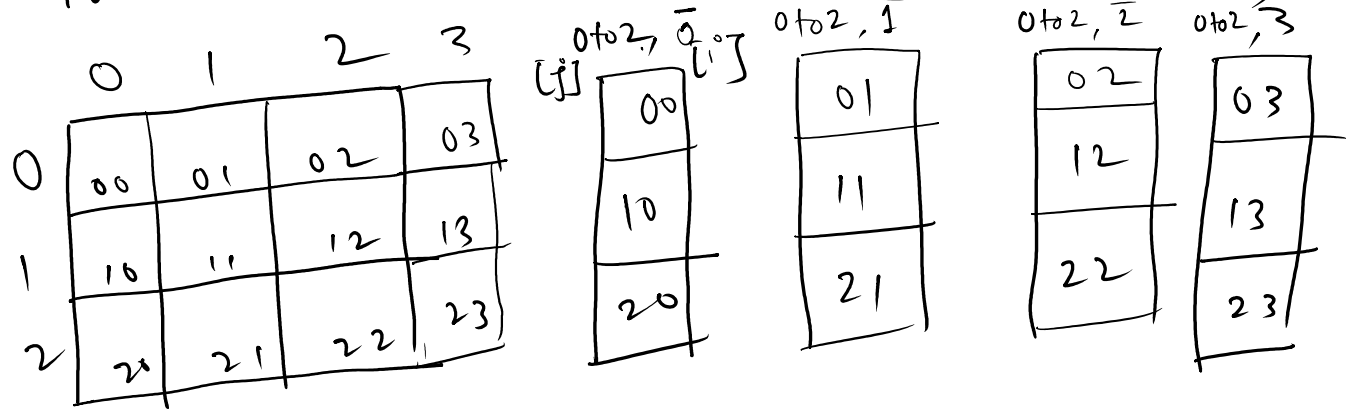
```
row = [1 1 0]
      0 1 2

col = [1 0 1 1]
      0 1 2 3
```

This gives information that which row and which col has 1.

3. Traverse entire matrix (Set all row with 1)

```
for(int i=0; i<row; i++) {
    if(row[i]==1) {
for(int j=0; j<col; j++) {
    arr[i][j]=1;
    }
}
```

4. Traverse the entire matrix (Set col with 1)




```
for(int i=0; i<col; i++) {
    if(col[i]==1) {
for(int j=0; j<row; j++) {
```

```
for(int j=0; j<row; j++) {
    mat[j][i] = 1;
}
}
```

Code:

```
int row[] = new int[row];
int col[] = new int[col];
for(int i=0; i<row; i++) {
  for(int j=0; j<col; j++) {
    if(mat[i][j] == 1) {
      row[i] = 1;
      col[j] = 1;
    }
  }
for(int i=0; i<row; i++) {
  if(row[i] == 1) {
for(int j=0; j<col; j++) {
  mat[i][j] = 1;
```

```
        }
    }
}
// set col.
for(int i=0; i<col; i++){
    if(col[i]==-1){
        for(int j=0; j<row; j++){
            arr[j][i]=1;
        }
    }
}
```