

# Print row wise with condition

even  $\rightarrow$  left to right  
odd  $\rightarrow$  right to left

arr

even

odd

	0	1	2	3	4
0	1	2	3	4	5
1	7	8	9	10	11
2	13	14	15	16	17
3	18	19	20	21	22
4	23	24	25	26	27
5	28	29	30	31	32

(m $\times$ n)

ans

1	2	3	4	5
11	10	9	8	7
13	14	15	16	17
22	21	20	19	18
23	24	25	26	27
32	31	30	29	28

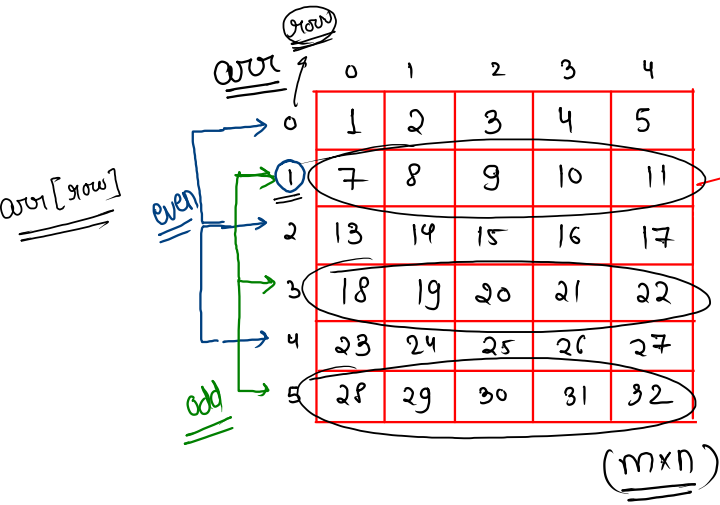


when only print  
(only for hacker rank) ( जुगाड़ )

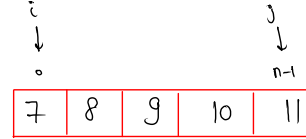
```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int m = scn.nextInt();
    int n = scn.nextInt();
    int[][][] arr = new int[m][n];
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            arr[i][j] = scn.nextInt();
        }
    }
    printRowWise(arr, m, n);
}

public static void printRowWise(int[][][] arr, int m, int n) {
    for (int i = 0; i < m; i++) {
        if ( i % 2 == 0 ) {
            for (int j = 0; j < n; j++) {
                System.out.print( arr[i][j] + " " );
            }
        } else {
            for (int j = n - 1; j >= 0; j--) {
                System.out.print( arr[i][j] + " " );
            }
        }
        System.out.println();
    }
}
```

logic



$5 \times 6 = 30 / 2 = 15$



```
int i=0, j=n-1;
while( i < j ) {
    swap( arr[i], arr[j] );
    i++;
    j--;
}
```

Code

```
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int m = scn.nextInt();
    int n = scn.nextInt();
    int[][] arr = new int[m][n];
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            arr[i][j] = scn.nextInt();
        }
    }
    int[][] ans = printRowWise(arr, m, n);
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            System.out.print(ans[i][j] + " ");
        }
        System.out.println();
    }
}

public static int[][] printRowWise(int[][] arr, int m, int n) {
    for (int row = 0; row < m; row++) {
        if (row % 2 != 0) {
            int i = 0;
            int j = n - 1;
            while (i < j) {
                swap(arr[row], i, j);
                i++;
                j--;
            }
        }
    }
    return arr;
}

public static void swap(int[] arr, int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}
```

operations :-  $\frac{m * n}{2}$

T.C =  $O\left(\frac{m * n}{2}\right)$

$\approx O(m * n)$

S.C =  $O(1)$

# Convert 1-D Array to 2-D Array

arr =

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Diagram showing the 1-D array divided into three segments of 5 elements each, indicated by brackets below the array.

$p = 3$  // rows

$q = 5$  // cols

(3 x 5)

ans

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15

## Observation

<u>index of 1D array element</u>		<u>index of 2D array element</u>	
<u>(idx)</u>	0	→	(0,0)
<u>(given)</u>	1	→	(0,1)
	2	→	(0,2)
	3	→	(0,3)
	4	→	(0,4)
	5	→	(1,0)
	6	→	(1,1)
	7	→	(1,2)
	8	→	(1,3)
	9	→	(1,4)
	10	→	(2,0)
	11	→	(2,1)
	12	→	(2,2)
	13	→	(2,3)
	14	→	(2,4)

(i,j)  
(found  
out)

## Formula

V.V  
(Imp)

$$\begin{aligned} i &= \text{idx} / q \\ j &= \text{idx} \% q \end{aligned}$$

(to find 2D index using 1D index)


Note:-

$$\text{idx} = i * q + j$$

(to find 1D index using 2D indexes)

Code

$$T.C = O(n) \text{ or } O(p*q)$$
$$S.C = O(p*q)$$

```
public static void convert1dTo2d(int[] arr, int n, int p, int q) {  
    int[][] ans = new int[p][q];  
    for (int idx = 0; idx < n; idx++) {  
        int i = idx / q;  
        int j = idx % q;  formula  
        ans[i][j] = arr[idx];  
    }  
  
    // print  
    for (int i = 0; i < p; i++) {  
        for (int j = 0; j < q; j++) {  
            System.out.print(ans[i][j] + " ");  
        }  
        System.out.println();  
    }  
}
```