Problem Statement – Wave Form Traversal

In the kingdom of **NumMatrix**, the Royal Messenger must deliver letters by walking through the castle halls. The castle is represented as an **N × M matrix**, where each room has a number.

The messenger follows a **wave-like path**:

- In the **first column**, he moves **top to bottom**.
- In the **second column**, he moves **bottom to top**.
- In the **third column**, again **top to bottom**.
- This continues for all columns.

Your task is to print the **wave form traversal** of the matrix.

Input Format

1. The first line contains two integers N and M – the number of rows and columns.
2. The next N × M numbers represent the matrix elements.

Output Format

- Print the elements of the matrix in **wave order traversal** (space-separated).

Constraints

- $1 \leq N, M \leq 100$
- $1 \leq arr[i][j] \leq 10^4$

Example 1

Input

```
3 4
1  2  3  4
5  6  7  8
9 10 11 12
```

Output

```
1 5 9 10 6 2 3 7 11 12 8 4
```

Example 2

Input

```
4 3
1  2  3
4  5  6
7  8  9
10 11 12
```

Output

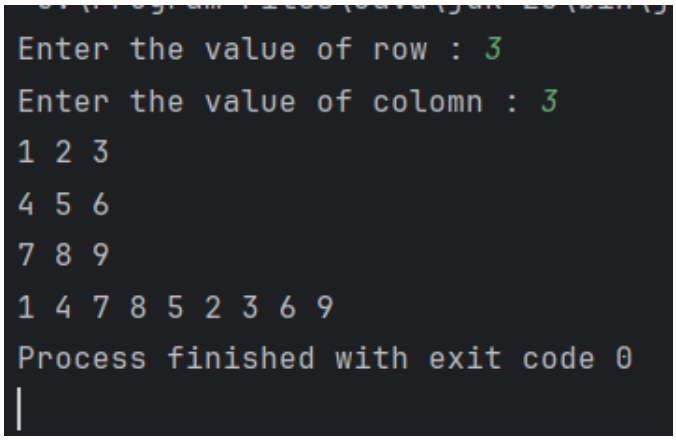
```
1 4 7 10 11 8 5 2 3 6 9 12
```

CODE :-

```java
import java.util.Scanner;

public class Wave_Transversal {
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.print("Enter the value of row : ");
        int N= sc.nextInt();
        System.out.print("Enter the value of colomn : ");
        int M=sc.nextInt();

        int mat[][]= new int [N][M];
        int k=1;
        for (int i=0; i<N; i++){
            for(int j=0; j<M; j++){
                mat[i][j]=k;
                k++;
            }
        }
//          FOR PRINTING THE MATRIX

        for(int i=0; i<N; i++){
            for(int j=0; j<M; j++){
                System.out.print(mat[i][j]+" ");
            }
            System.out.println();
        }
```

```java
//          REAL LOGIC

        for( int j=0; j<M; j++){
            if(j%2==0){
                for(int i=0; i<N; i++){
                    System.out.print(mat[i][j]+" ");
                }
            }else{
                for( int i=N-1; i>=0; i--){
                    System.out.print(mat[i][j]+" ");
                }
            }
        }

    }
}
```

OUTPUT :-



Problem Statement – Transpose of a Matrix

In the **Academy of NumMatrix**, students are given a magical board represented as an **N × M matrix**.
The headmaster wants to flip the matrix along its **main diagonal** so that rows become columns and columns become rows.
This operation is called a **Transpose**.

Your task is to help the headmaster by printing the **transpose of the given matrix**.

Input Format

1. The first line contains two integers N and M – the number of rows and columns.
2. The next N × M numbers represent the matrix elements.

Output Format

- Print the **transpose matrix** (of size M × N).

Constraints

- 1 ≤ N, M ≤ 100
- 1 ≤ arr[i][j] ≤ 10^4

Example 1

Input

```
3 3
1 2 3
4 5 6
7 8 9
```

Output

```
1 4 7
2 5 8
3 6 9
```

Example 2

Input

```
2 3
1 2 3
4 5 6
```

Output

```
1 4
2 5
3 6
```

CODE :-

```java
import java.util.Scanner;

public class TransposeInMatrix {
    public static void main(String[] args) {
        Scanner sc =new Scanner(System.in);
        int n=sc.nextInt();
        int arr[][]=new int[n][n];

        int k=1;
//        printing the matrix
        System.out.println("matrix");
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                System.out.print(k+" ");
                arr[i][j]=k;
                k++;
            }
            System.out.println();
        }
```

```java
//        Tranpose
        int arr1[][]=new int[n][n];
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                arr1[j][i]=arr[i][j];
            }
        }
//        Printing the tranpose Matrix
        System.out.println("transpose");
        for(int i=0; i<n; i++){
            for(int j=0; j<n; j++){
                System.out.print(arr1[i][j]+" ");
            }
            System.out.println();
        }

    }
}
```

OUTPUT :-

```
3
matrix
1 2 3
4 5 6
7 8 9
transpose
1 4 7
2 5 8
3 6 9

Process finished with exit code 0
```

Problem Statement – Spiral Traversal of a Matrix

In the **Royal Garden of NumMatrix**, the King wants to enjoy the flowers arranged in an **N × M rectangular layout**.
He instructs his gardener to walk in a **spiral path** starting from the **top-left corner**:

1. Walk **left to right** along the top row.
2. Then walk **top to bottom** along the rightmost column.
3. Then walk **right to left** along the bottom row.
4. Then walk **bottom to top** along the leftmost column.
5. Continue the process inward until every element is visited.

Your task is to print the **spiral order traversal** of the given matrix.

Input Format

1. First line: Two integers N and M (rows and columns).
2. Next N × M integers: The elements of the matrix.

Output Format

- Print the matrix elements in **spiral order traversal** (space-separated).

Constraints

- $1 \leq N, M \leq 100$
- $1 \leq arr[i][j] \leq 10^4$

Example 1

Input

3 3
1 2 3
4 5 6
7 8 9

Output

1 2 3 6 9 8 7 4 5

Example 2

Input

3 4
1 2 3 4
5 6 7 8
9 10 11 12

Output

1 2 3 4 8 12 11 10 9 5 6 7

CODE :-

```java
import java.util.*;

public class Spiral_Matrix {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter the size of matrix (n x m)");
        System.out.print("Enter the value of n : ");
        int n = sc.nextInt();
        System.out.print("Enter the value of m : ");
        int m = sc.nextInt();

        int matrix[][] = new int[n][m];

        System.out.println("Enter matrix elements:");
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                matrix[i][j] = sc.nextInt();
            }
        }

        System.out.println("The spiral order of matrix is: ");
        int rowStart = 0;
        int rowEnd = n - 1;
        int colStart = 0;
        int colEnd = m - 1;
```

```
        while (rowStart <= rowEnd && colStart <= colEnd) {

            // Top row
            for (int col = colStart; col <= colEnd; col++) {
                System.out.print(matrix[rowStart][col] + " ");
            }
            rowStart++;

            // Right column
            for (int row = rowStart; row <= rowEnd; row++) {
                System.out.print(matrix[row][colEnd] + " ");
            }
            colEnd--;

            // Bottom row (reverse)
            if (rowStart <= rowEnd) {
                for (int col = colEnd; col >= colStart; col--) {
                    System.out.print(matrix[rowEnd][col] + " ");
                }
                rowEnd--;
            }

            // Left column (reverse)
            if (colStart <= colEnd) {
                for (int row = rowEnd; row >= rowStart; row--) {
                    System.out.print(matrix[row][colStart] + " ");
                }
                colStart++;
            }
        }
    }
}
```

OUTPUT :-

```
Enter the size of matrix (n x m)
Enter the value of n : 3
Enter the value of m : 3
Enter matrix elements:
1 2 3
4 5 6
7 8 9
The spiral order of matrix is:
1 2 3 6 9 8 7 4 5
Process finished with exit code 0
```