Multi-Dimensional array & Wrapper classes

1. Write a program to print 2D matrices

```
    import java.util.Scanner;

public class _2DArray_Matrix {
          public static void main(String[] args) {


                    Scanner sc = new Scanner(System.in) ;
                int raw  = sc.nextInt();
                int col  = sc.nextInt();

              int[][] matrix = new int[raw][col];

              //Input
              for(int i = 0; i<raw ;i++) {
                    for ( int j=0; j<col ;j++) {
                              matrix[i][j]  = sc.nextInt();
                    }
              }


              // Output
              for(int i = 0; i<raw ;i++) {
                    for ( int j=0; j<col ;j++) {
                              System.out.print( matrix[i][j] + " ");
                    }
                    System.out.println();
              }
          }
          }
```

2. Write a program to subtract two matrices

```
   public class MatrixSubtraction {

   public static void main(String[] args) {
      // Example matrices
```

```
        int[][] matrix1 = {
            {5, 7, 9},
            {2, 4, 6},
            {8, 1, 3}
        };

        int[][] matrix2 = {
            {3, 2, 1},
            {6, 5, 4},
            {9, 8, 7}
        };

        // Subtract matrices
        int[][] result = subtractMatrices(matrix1, matrix2);

        // Display the result
        System.out.println("Matrix 1:");
        printMatrix(matrix1);

        System.out.println("\nMatrix 2:");
        printMatrix(matrix2);

        System.out.println("\nSubtraction Result:");
        printMatrix(result);
    }

    // Method to subtract two matrices
    private static int[][] subtractMatrices(int[][] mat1, int[][] mat2) {
        int rows = mat1.length;
        int columns = mat1[0].length;

        int[][] result = new int[rows][columns];

        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                result[i][j] = mat1[i][j] - mat2[i][j];
            }
        }

        return result;
    }

    // Method to print a matrix
    private static void printMatrix(int[][] matrix) {
        for (int[] row : matrix) {
            for (int element : row) {
                System.out.print(element + " ");
            }
            System.out.println();
        }
    }
}
```

3. Write a program print the Zigzag 2D Matrix

```
public class _2D_Array_ZigZag {

    public static void main(String[] args) {

        int[][] mat = {{1,2,3,4,5}, {6,7,8,9,1}, {3,2,5,4,6}, {7,8,9,1,2}};

        int n = mat.length;
        int m = mat[0].length;

        ZigzagTraverse(mat, n, m);
    }

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

}
```

4. Write a program print the Spiral ClockwiseTraversal of 2D Matrix

```
public class _2D_Array_SpiralClockwiseTraversal {
    public static void main(String[] args) {
        int[][] matrix = {
            {1, 2, 3, 4},
            {5, 6, 7, 8},
            {9, 10, 11, 12},
            {13, 14, 15, 16}
        };

        int rows = matrix.length;
        int cols = matrix[0].length;

        int top = 0;
        int bottom = rows - 1;
        int left = 0;
        int right = cols - 1;

        while (top <= bottom && left <= right) {
            // Traverse top row
            for (int i = left; i <= right; i++) {
                System.out.print(matrix[top][i] + " ");
            }
            top++;

            // Traverse rightmost column
            for (int i = top; i <= bottom; i++) {
                System.out.print(matrix[i][right] + " ");
            }
            right--;

            // Check if there is a row below
            if (top <= bottom) {
                // Traverse bottom row
                for (int i = right; i >= left; i--) {
                    System.out.print(matrix[bottom][i] + " ");
                }
                bottom--;
            }

            // Check if there is a column on the left
            if (left <= right) {
                // Traverse leftmost column
                for (int i = bottom; i >= top; i--) {
                    System.out.print(matrix[i][left] + " ");
                }
                left++;
            }
        }
    }
}
```

5. Write a program print the sum of swastik 2D matrix

```
public class _2D_Array_SwastikSum {

    public static void main(String[] args) {


        int row = 3;
        int col = 5;

        int mat[][] = {{1,2,3,4,5},
                       {6,7,8,9,10},
                       {11,12,13,14,15}
                      };

        int sum1 = 0;
        int sum2 = 0;

        // sum1

        for( int j =0 ; j <= (col-1)/2 ; j++ ){
            sum1 = sum1 + mat[row-1][j] ;

        }


        for( int i = row-2 ; i>=0  ; i-- ){
            sum1 = sum1 + mat[i][(col-1)/2 ] ;

        }

        for( int j =((col-1)/2 )+1  ;j < col  ; j++){
            sum1 = sum1 + mat[0][j] ;
        }
    System.out.println(sum1);



    // sum2

    for( int i =0 ; i <= (row-1)/2 ; i++ ){
            sum2 = sum2 + mat[i][0] ;
        }

        for( int j = 1 ; j<col ; j++){
            sum2 = sum2 + mat[(row-1)/2 ][j] ;
        }

        for( int i =((row-1)/2 )+1 ;i < row  ; i++){
            sum2 = sum2 + mat[i][col-1] ;
        }

        System.out.println(sum2);

        int difference = sum1 - sum2;
        int ans = Math.abs(difference);

        System.out.println(ans);

    }

}
```

6. Write a program print the elements of the matrix, row by row, by adding 1 to each element

```
import java.util.Scanner;

public class _2D_Array_Add1InMatrix {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);
```

```
        // Input the dimensions of the rectangular matrix
        int n = scanner.nextInt(); // Number of rows
        int m = scanner.nextInt(); // Number of columns

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

        // Input the elements of the matrix
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                matrix[i][j] = scanner.nextInt();
            }
        }

        // Increment each element of the matrix by 1 and print the updated matrix
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                matrix[i][j] += 1;
                System.out.print(matrix[i][j] + " ");
            }
            System.out.println(); // Move to the next row
        }
   scanner.close();
    }
}
```

7. Java program to illustrate the concept of Autoboxing

```
public class Autoboxing {

    public static void main(String[] args) {
        // Autoboxing: converting primitive int to Integer
        int primitiveInt = 42;
        Integer wrappedInt = primitiveInt;

        // Autoboxing: converting primitive double to Double
        double primitiveDouble = 3.14;
        Double wrappedDouble = primitiveDouble;

        // Displaying the values
        System.out.println("Wrapped Integer: " + wrappedInt);
        System.out.println("Wrapped Double: " + wrappedDouble);

    }
}
```

8.Java program to illustrate the Concept of Unboxing

```
public class UnboxingExample {

    public static void main(String[] args) {

        // Autoboxing: converting primitive int to Integer

        Integer wrappedInt = 42;



        // Autoboxing: converting primitive double to Double

        Double wrappedDouble = 3.14;



        // Autoboxing: converting primitive boolean to Boolean

        Boolean wrappedBoolean = true;



        // Autoboxing: converting primitive char to Character

        Character wrappedChar = 'A';



        // Unboxing: converting Integer to primitive int
        int primitiveInt = wrappedInt;

        // Unboxing: converting Double to primitive double
        double primitiveDouble = wrappedDouble;

        // Unboxing: converting Boolean to primitive boolean
        boolean primitiveBoolean = wrappedBoolean;

        // Unboxing: converting Character to primitive char
        char primitiveChar = wrappedChar;

        // Displaying the values
        System.out.println("Unboxed Integer: " + primitiveInt);
        System.out.println("Unboxed Double: " + primitiveDouble);
        System.out.println("Unboxed Boolean: " + primitiveBoolean);
        System.out.println("Unboxed Character: " + primitiveChar);
    }
}
```

By - Divya Parihar