4. n-dimensional arrays

Note: all your methods should be `public static` in this module
Resources:

n-dimensional arrays

Problems

1. Write a method that takes a two-dimensional array of integers as its input. The method should print every integer in the array.

2. Write a method that takes a two-dimensional array of integers as its input. The method should print every integer in every row with an odd index.

   Example behavior:

   Input: `double[][] a = { {1, 2}, {3, 4}, {6, 7}}`
   Output: `3, 4`

3. Write a method that takes a three-dimensional array of integers as its input and returns the sum of every integer in the array.

4. Write a method that takes two-dimensional array of integers as its input. The method should determine whether the two-dimensional array is shaped like a square.

   Example behavior:

   Input: `double[][] a = { {1.1, 2}, {3, 4.5}}`
   Output: `true`

   Input: `double[][] a = { {1.1, 2}, {3}}`
   Output: `false`

5. Write a method that multiplies a matrix by a scalar.

   Example behavior:

   Input: `double scalar, double[][] matrix`
   Output: the calculated matrix `double[][]`

6. Write a method that multiplies a matrix by a vector. You should return an empty vector when the input is invalid.

   Example behavior:

   Input: `double[] vector, double[][] matrix`
   Output: the calculated vector `double[]`

7. Write a method that calculates the sum of two matrices. You should return an empty matrix when the input is invalid.

   Example behavior:

   Input: `double[][] a, double[][] b`
   Output: the sum of matrix `a` and matrix `b` as `double[][]`