CSD 3464 – ASSIGNMENT 04 (Question 1)

Overview:

The following question is **not** a question from *Absolute Java* (6th Ed.). Please follow the instructions included in **this** document and implement the following Java files:

⇒ ArrayUtilities2D
⇒ ArrayUtilities2DTester.java *(Contains a `main()` method)*

The above classes/files should be inside a package called q*1*.

Instructions:

In this question you are to develop a set of `static` utility `methods` that take one or more 2D arrays as parameters. These methods will reside in the class `ArrayUtilities2D` and this class should **NOT** have any non-static fields/instance variables.

The first `static` method you are tasked with developing is a `sumArray` which takes a single `array2D` of type `double[][]` as a parameter returns a `double` value that stores the resulting of summing all the values in the `array2D`.

Similarly, you must develop a `static` method named `multiplyArray` which takes a single `array2D` of type `double[][]` as a parameter and returns a `double` value that stores the result of multiplying all the values in the `array2D`.

Next, you need to develop a static method named `searchArray` that takes a `array2D` of type `double[][]` and `SearchCriteria` object and returns the maximum or minimum value of the `array2D` depending on the selected `SearchCriteria`.

- Calling `searchArray(array2D, ArrayUtilities2D.SearchCriteria.MAX)` should return the maximum value in `array2D`
- Calling `searchArray(array2D, ArrayUtilities2D.SearchCriteria.MIN)` should return the minimum value in `array2D`

Another `static` method your class must contain is an `equals` method that takes two 2D arrays of type `double[][]` as parameters and returns a `boolean` value indicating if the two 2D arrays have the same contents.

- The two 2D arrays cannot be equal if they are of different sizes, both in terms of number of rows and number of columns
- If both 2D arrays are of equal size, they are only equal if their contents are the same for each index.

The fifth `static` method you are to develop is `sumColumn` which takes a single `array2D` of type `double[][]` and a `int` indicating the `col` of the `array2D` you would like to sum. This method should return an `int` value storing the sum of the requested column, if the column number provided is invalid return `0`.

- Assume calling `sumColumn(array2D, 0)` would sum the first column
- Ensure in the method a valid column number is selected, a value less than 0 or greater than the number of columns in `array2D` minus one are invalid.
    - Return `0` from the method in the above cases

The sixth `static` method you are to develop is `sumRow` which takes a `array2D` array of type `double[][]` and a `int` indicating the `row` of the `array2D` you would like to sum. This method should return an `int` value storing the sum of the requested row, if the row number provided is invalid return `0`.

- Assume calling `sumRow(array2D, 0)` would sum the first row
- Ensure in the method a valid row number is selected, a value less than 0 or greater than the number of rows in `array2D` minus one are invalid.
    - Return `0` from the method in the above cases

You are also required to write a class called `ArrayUtilities2DTester` in another file that contains a `main()`. The purpose of this class is to test the functionality of your `ArrayUtilities2D` class. It is up to you to decided what to include in the `main()` to test the class; however, all `public` facing methods need to be tested for correctness.

The basic structure of your `ArrayUtilities2D` class is required to look like the following:

```
/**
 * Utility class to work with 2D arrays
 */
public class ArrayUtlities2D
{

     public static enum SearchCriteria {MAX, MIN};

     /**
      * Calculate sum of all elements in a two-dimensional array
      * @param arrray2D Two-dimensional array storing double values
      * @return Sum of all elements inside provided 2D array
      * frequency in given word
      */
     public static double addArray(double[][] array2D)
     {
        // insert code here
     }
```

```java
/**
 * Return the result of multiplying all elements in a two-
 * dimensional array
 * @param arrray2D Two-dimensional array storing double values
 * @return Result of multiplying all elements inside provided 2D
 *         array
 */
public static double multiplyArray(double[][] array2D)
{
    // insert code here
}

/**
 * Returns the max/min value from the provided 2D array
 * @param array2D Two-dimensional array containing double values
 * @param criteria Value from SearchCriteria enum indicating if
 * you are searching for MAX or MIN value in 2D array
 * @return Maximum/minimum value of two-dimensional array
 */
public static double searchArray(double[][] array2D,
      SearchCriteria criteria)
{
    // insert code here
}

/**
 * Returns a boolean value indicating if two two-dimensional
 * arrays are equal. Arrays are equal if they are of the same
 * size and contain the same contents.
 * @param array2dOne First 2D array containing doubles
 * @param array2dTwo Second 2D array containing doubles
 * @return true if two arrays are equal, false otherwise
 */
public static boolean equals(double[][] array2dOne,
      double[][] array2dTwo)
{
    // insert code here
}

/**
 * Returns the sum of the specified column should the column
 * exist, returns 0 otherwise
 * @param array2D Two-dimensional array containing double values
 * @return Sum of all the elements in the specified column of the
 *         2D array
 */
public static double sumColumn(double[][] array2D, int col)
{
    // insert code here
}
```

```
    /**
     * Returns the sum of the specified row should the row
     * exist, returns 0 otherwise
     * @param array2D Two-dimensional array containing double values
     * @return Sum of all the elements in the specified row of the
     *         2D array
     */

    public static double sumRow(double[][] array2D, int row)
    {
        // insert code here
    }

}
```