Contents

Two-Dimensional Arrays

Solutions for those exercises.

Multiple Choices

1. What is the correct way of creating a 2-dimensional rectangular array of `int` with 5 rows and 2 columns named `myMatrix`?

   ☐ `int[][] myMatrix = new int[5][2];`
   ☐ `int[][] myMatrix = new int[2][5];`
   ☐ `int[,] myMatrix = new int[2, 5];`
   ☐ `int[,] myMatrix = new int[5, 2];`

2. Consider the following code:

   ```
   int[,] grades = {{10, 20}, {30, 40}};
   Console.WriteLine(grades[1,0]);
   ```

   What will it display?

   ☐ Nothing
   ☐ 10
   ☐ 20
   ☐ grades
   ☐ 30
   ☐ grades(1,0)
   ☐ 40

Exercises

1. Write a statement that creates a 2-dimensional rectangular array of `int` with 5 rows and 3 columns.

2. Write a statement that creates a 2-dimensional jagged array of `int` with 2 rows. The first row should contain an array containing 1, the second row should contain an array containing 2, 3.

3. Write a declaration for a 2-dimensional rectangular array of `int` containing the following data:

10	20	30
40	50	60
70	80	90

4. Write a declaration for a 2-dimensional ragged array of `int` containing the following data:

10	20	
40		
70	80	90

5. Suppose we have a 2-dimensional rectangular array named `temp` that has been declared and initialized. How can we know the number of rows in this array?

6. Suppose we have a 2-dimensional rectangular array named `temp` that has been declared and initialized. How can we know the number of elements in this array?

7. Write a `Display` static method that takes as an argument a 2-dimensional array and displays it at the screen.

8. Write a program that display "Every row contains its own index" if the 2-dimensional rectangular array of `int matrix` is such that its first row contains the value 0, its second row contains the value 1, etc.

9. Write a program that display the average of each row of a 2-dimensional jagged array of `int jArray`.

10. Write a program that display the sum of the values on the diagonal of a 2-dimensional rectangular array of `int jArray`.

11. Write a program that "rotate" a 2-dimensional array 90° clockwise. For example, the array

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

```
        { 10, 11, 12 },
    };
```

would become

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

Wrapping-up Problem

1. Answer the following questions using a 2-dimensional jagged array **and** a 2-dimensional rectangular array.

   (a) Write statements that declare a 2-dimensional array with at least 2 rows containing values 1 through 6.

   (b) Write statements that display the values stored in a 2-dimensional array called `arTest`.

   (c) Write statements displaying "Ranked" if the sum of the values stored in each row is greater than the sum of the values in the preceding row in the `arTest` array. For example, the following rectangular and jagged arrays are both ranked, as 4 < 5 < 6.

   ```
   2 1 1
   3 1 1
   5 1 0

   4 0
   3 1 1
   2 2 1 1
   ```

Problem: Toward a Crossword Puzzle Solver

The goal of this problem is to work toward the creation of a program that solve crossword puzzles. We will reason in the simple case where the "word" is actually simply a pair of number (so, "1, 2" or "8, 101").

In the following, assume given two `int` variables `first` and `second`, as well as a 2-dimensional rectangular array `values`.

1. Write a program that display "pair found" if `first` and `second` occur next to each other in the same row.
2. Edit your program so that "pair found" is displayed also if `second` occurs before `first` in the same row.
3. Edit your program so that "pair found" is displayed also if `first` occurs "above" `second` (that is, if they are next to each other in the same column),

4. Edit your program so that "pair found" is displayed also if `second` occurs "above" `first`,
5. Edit your program so that "pair found" is displayed also if `first` and `second` occur diagonally,
6. Edit your program so that "pair found" is displayed also if `first` and `first` occur anti-diagonally.

Test your program thoroughly, possibly bundling it in a **static** class to ease testing and debugging.