Multidimensional Arrays

Complete the questions below with the aid of your textbook, notebook, or classroom resources. You must work on the problem on your own. When your instructor calls “time,” pass your paper to the person next to you, and accept the new worksheet. You should first work on the next blank problem on the sheet, then, if you have time, scan through the sheet to see if you spot any errors in others’ work. Correct those errors if you have time, then pass the sheet again the next time your instructor calls “time.”

1. Write code that correctly initializes an array that holds 1028 values of type double.
2. Write code that correctly declares a two dimensional array with 5 rows and 4 columns.
3. Write code to construct an array called *studentSeating* that will be used as a seating chart for the classroom. You do not need to fill in any of the elements of the array, and you can assume that the final array will have empty seats (you do not need to construct a jagged array).
4. Draw a picture that illustrates what the memory looks like for the array *studentSeating*. Don’t forget to include the values that have been auto-initialized!
5. Write code that fills in the elements of the array from *studentSeating* with the names of students seated in each spot.
6. If you wanted to print the name of the person in the last desk of the second row, what code would you write?
7. Teachers get tired being on their feet all day! Write a line of code that would assign a seat for your teacher.
8. It’s your lucky day! You get to switch seats with any one student in the class. Write the code that will switch seats for you and the other student in the *studentSeating* array.
9. Your cousin is in charge of cooking pernil for your family holiday party. It’s her first time doing it, and so she takes the temperature of the pernil two times an hour, for all 10 hours (she’s really nervous!). Write a declaration for a two dimensional array to store all of the temperatures she collects.
10. Write a two dimensional String array declaration that would represent a chess board (a chess board is 8 spaces by 8 spaces). Pawns occupy all spaces in the first and 7th row of a chess board when the game first starts. Write code to fill those spaces with the word “pawn.”
11. Assume that a two-dimensional rectangular array of integers called *data* has been declared with four rows and seven columns. Write a loop to initialize the third row of data to store the numbers 1 through 7.
12. Assume that a two-dimensional rectangular array of integers called matrix has been declared with six rows and eight columns. Write a loop to copy the contents of the second column into the fifth column.
13. Consider the following method:

public static void mystery(int[] [] a){

for(int r = 0; r < a.length; r++){

for(int c = 0; c < a[0].length – 1; c++){

if (a[r][c + 1] > a[r][c]){

a[r][c] =a[r][c+1};

}

}

}

}

If a two-dimensional array numbers is initialized to store the following integers, what are its contents after the call shown below:

int[][] numbers = {{3, 4, 5, 6}, {4, 5, 6, 7], [5, 6, 7, 8]};

mystery(numbers);

1. Declare, fill, and print a 10 x 12 array that we could give to the elementary classes to help students learn their multiplication tables.

*Hint: it might help to draw the output first, and work your way backwards. Include comments and pseudocode to explain your program to others.*

1. The elementary teachers are probably going to want to test their students on their multiplication tables. Write code that will copy the contents of the third column into the seventh column, so students will have to find and correct errors in the table.
2. Write a special multiplication table that only includes even numbers multiplied by other even numbers. Your first row should read {2, 4, 6, 8, 10}.
3. Adjust the special multiplication table so that it only includes even number multiplied by odd numbers. Your first row should read {2, 4, 6, 8, 10}.
4. Write code that constructs an array called *magicBox* that outputs the following:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 1 | 2 | 2 | 2 | 1 | 0 |
| 0 | 1 | 2 | 3 | 2 | 1 | 0 |
| 0 | 1 | 2 | 2 | 2 | 1 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Your code should accept a parameter so you can make an array of any desired dimensions.

1. Write a code that constructs an array called *pyramid* that outputs the following array:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 2 | 2 | 2 | 2 | 2 |
| 0 | 1 | 2 | 3 | 3 | 3 | 3 |
| 0 | 1 | 2 | 3 | 4 | 4 | 4 |
| 0 | 1 | 2 | 3 | 4 | 5 | 5 |
| 0 | 1 | 2 | 3 | 4 | 5 | 5 |