AP Computer Science
Homework Set 6
2D Arrays

Note: all programs described below should work if the size of the 2D array is changed.

P6A. Create a 3x4 2D array of integers and fill it with random numbers between 0 and 9. Use a nested for loop to print the array in a "rectangular" format. Finally, print the row, column pair of coordinates of all instances of the number "5". Use nested for loops to "populate the 2D array" and print out its contents. The strategic use of if statements and "tabs" will allow you to print the array in a "rectangular" format (i.e. 3 rows "deep" and 4 columns "wide")

P6B. Create a 3x5 2D array of integers called "`numbers`" and fill it with numbers 1-15 in row-major (left to right, top to bottom) order using a nested **for loop**. If the dimensions of "`numbers`" are changed, the numbers filling "`numbers`" should also change to fill the array. For instance, if the array dimensions are 6x7, then the array will be filled with numbers 1-42.

a. Print the 2D array "`numbers`" in "rectangular" format using nested **for-each loop**.

b. Complete the following computations:
   i. Calculate and print the sum total of all 15 elements of the array.
   ii. Calculate and print the sum total of each row in the array.
   iii. Calculate and print the sum total of each column in the array.

   Preface each sum with text describing which sum is being printed.

c. Print the 2D array "numbers" in column-major order. Note: the numbers originally placed into "numbers" should remain intact, i.e., they still exist in row-major order. Only the printing process should be in column-major order.

P6C. This program will allow us to transfer elements from 1D arrays 2D arrays. The program should work for any 1D length or 2D array dimensions.

1D Array

a. Create a 1D array called "`even1D`" that can hold 10 integers.
b. Populate the 1D array with even numbers beginning with `0` using a `for loop`. If the length of "`even1D`" is changed, the algorithm for filling "`even1D`" should continue to completely fill "`even1D`" with even numbers.
c. Print the contents of "`even1D`" using a `for-each` loop.

2D Array of even numbers

d. Create a 2D array called "`even2D`" with dimensions 4 x 5.
e. Copy the values from "`even1D`" into "`even2D`" in **<u>row-major</u>** order. Here are three cases to consider:
    i. if the number of elements in "`even1D`" array exactly matches the number of elements in "`even2D`", there should be no "blank" spaces in the array.
    ii. if the number of elements in "`even1D`" is LESS than the number of elements in "`even2D`", there will be blank spaces. These "blank" spaces should be filled in with the value "-1".
    iii. if the number of elements in the 1D array is GREATER than the number of elements in "`even2D`", only those values that "fit" into "even2D" should be filled.
f. Print the contents of "`even2D`" using either a `for loop` or `for-each` loop in "rectangular format".

P6D. 2015 #1 DiverseArray (handwritten)

P6E. 2014 #3 Seating Chart (handwritten)

P6F. 2016 #3 Crossword (handwritten)

P6G. 2015 #3 SparseArray (handwritten)

P6H. Below is the class definition for class Array2D. (handwritten)

```
public class Array2D
   {
   private int[][] numbers;

   public Array2D()
      {
      numbers = new int[][]  { {  1,  2,  3 },
                               {  4,  5,  6 },
                               {  7,  8,  9 } };
      } // end zero-arg constructor


   public List<Integer> toArrayList()
      {

      // returns a ArrayList of values in the 2D array
      // in "snaking" order

      } // end method toArrayList
   } // end class Array2D
```

Write the method `toArrayList()` that will return an `ArrayList` of the Integer values in the 2D array in “snaking order”. Snaking order is defined to be traversing the first row in row-major order, then continuing on to the next row at `numbers.length` (the last column), and continuing to the next row to column = 0.

A sample 2D array is shown below:

1	**2**	**3**
4	**5**	**6**
7	**8**	**9**

The `toArrayList()` method should return the following `ArrayList`:

1 2 3 6 5 4 7 8 9

When complete, you may test your code in the template.

Learning Objective Checklist

(please print and complete after you have had all HW Set 6 programs checked off)

Place a check next to those items that you have mastered

a. Write the Java code to create and populate a 2D array.

b. Write nested for and for-each loops to populate and process 2D arrays.

c. Write nested for and for-each loops to traverse 2D arrays in row-major and column-major order.