Introduction to Computer Science 1 – Homework 5

1.(on paper) Fill in the array with the values that would be stored after the code executes.

```
int[] data = new int[8];
data[0] = 3;
data[7] = -18;
data[4] = 5;
data[1] = data[0];
int x = data[4];
data[4] = 6;
data[x] = data[0] * data[1];
```

0	1	2	3	4	5	6	7

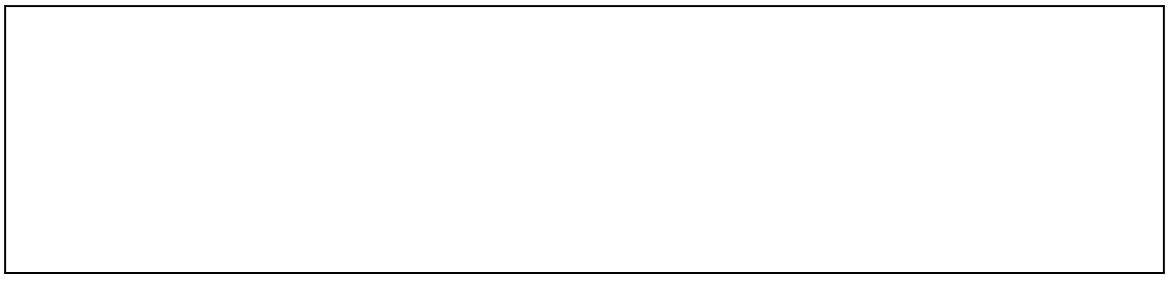
2.(on paper) Fill in the array with values that would be stored after the code executes.

```
int[] list = {2, 18, 6, -4, 5, 1};

for (int i = 0; i < list.length; i++) {

      list[i] = list[i] + (list[i] / list[0]);

}
```

0	1	2	3	4	5

3.(on paper) Draw the memory representation of the following arrays/statements.

```
boolean[] mary = {true, true, false};
boolean[] john;
boolean[] alice = null;
john=alice;
mary=john;
```



4.(on paper) Suppose we have the following program. What are the contents of array `a5` after the execution of mystery method?

```
public class ILoveArrays {
      public static void main(String[] args) {
            int[] a5 = {2,4,6,3,7,9};
            mystery(a5);
      }

      public static void mystery(int[] a) {
            for (int i = 0; i < a.length - 1; i++) {
                  if (a[i] < a[i + 1]) {
                    a[i] = a[i + 1];
                   }
            }
      }
}
```

5.(on paper) Find the errors (syntax, logical, semantic) in the following method which is supposed to take as parameter an array of integers and find and return the maximum value in that array.

```
public static int max(int data[10]) {
       int max = 0;

       for (int i = 0; i < data[].length(); i++){

        if (array[i] > max) {
             max = array[i];}
        }
       return max[];
}
```

6. Basic operations with arrays: Write a method that takes as *parameter* an integer array **a** and *returns* the average of its elements.

Other variations might include (try them at home):
-Return the maximum/minimum element
-Return the index (position) of the maximum/minimum element
-Return the element closest to zero
-Print all the numbers greater than the average
-Print the elements in increasing order

7. Write a method named `percentEven` that accepts an array of integers as a parameter and returns the percentage of even numbers in the array as a real number. For example, if a variable named `nums` refers to an array of the elements `{6, 2, 9, 11, 3}`, then the call of `percentEven(nums)` should return `40.0`. If the array contains no even elements or no elements at all, return `0.0`.

8. In order to know which student is passing/failing the course I keep a double array called `grail`. The size of the array is 15 (assumes we are a smaller class). Declare and initialize properly the array (with some arbitrary grades). Then, write

a method called `checkGrades` which takes as parameter the grail array and checks whether every single one of you passed (i.e. if all elements in the array are greater than 5.5). If yes, then your method should return true, otherwise (i.e. even if only one (or more) student fails) should return false.

9. Like Strings (and any other object), you cannot compare two arrays by writing a==b. There is a method .equals (same idea as Strings), but let's see how this is actually implemented (you will make it happen)! Write a method named `equals` that accepts two arrays of integers as parameters and returns true if they contain exactly the same elements in the same order, and false otherwise. Note that the arrays might not be the same length; if the lengths differ, return false.

For example, if the following arrays are declared:

```
int[] a1 = {10, 20, 30, 40, 50, 60};
int[] a2 = {10, 20, 30, 40, 50, 60};
int[] a3 = {20, 3, 50, 10, 68};
```

The call `equals(a1,a2)` returns true but the call `equals(a1,a3)` returns false.

10. Write a method named minGap that accepts an integer array as a parameter and returns the minimum 'gap' between adjacent values in the array. The gap between two adjacent values in a array is defined as the second value minus the first value. For example, suppose a variable called array is an array of integers that stores the following sequence of values:

```
int[] array = {1, 3, 6, 7, 12};
```

The first gap is 2 (3 - 1), the second gap is 3 (6 - 3), the third gap is 1 (7 - 6) and the fourth gap is 5 (12 - 7). Thus, the call of minGap(array) should return 1 because that is the smallest gap in the array. If you are passed an array with fewer than 2 elements, you should return 0.

11. Write a method called `append` that accepts two integer arrays as parameters and returns a new array that contains the result of appending the second array's values at the end of the first array.

12. Write a method named `copyRange` that takes as parameters two arrays `a1` and `a2`, two starting indexes `i1` and `i2`, and a length `l`, and copies the first `l` elements of `a1` starting at index `i1` into array `a2` starting at index `i2`.

For example, if the following arrays are declared:

```
int[] a1 = {10, 20, 30, 40, 50, 60};
int[] a2 = {91, 92, 93, 94, 95, 96};
copyRange(a1, a2, 0, 2, 3);
```

After the preceding call, the contents of `a2` would be {91, 92, 10, 20, 30, 96}. You may assume that the parameters' values are valid, that the arrays are large enough to hold the data, and so on.

13. Write a method named `swapAll` that accepts two arrays of integers as parameters and swaps their entire contents. You may assume that the arrays passed are not null and are the same length.

For example, if the following arrays are passed:

```
int[] a1 = {11, 42, -5, 27, 0, 89};
int[] a2 = {10, 20, 30, 40, 50, 60};
swapAll(a1, a2);
```

After the call, the arrays should store the following elements:

```
a1: {10, 20, 30, 40, 50, 60}
a2: {11, 42, -5, 27, 0, 89}
```

14. Write a method called `longestSortedSequence` that accepts an array of integers as a parameter and that returns the length of the longest sorted (non-decreasing) sequence of integers in the array.

15. In the final round of a beauty contest, the performance of the final 5 contestants is judged by 3 judges. Write a program that allows a user to input the scores by 3 judges for each contestant. Then you should print: the total score of each contestant and the average score of all the five contestants.

16. Given a 9-by-9 array of integers between 1 and 9, check if it is a valid solution to a Sudoku puzzle: each row, column, and block should contain the 9 integers exactly once.

```
 5 3 4 | 6 7 8 | 9 1 2
 6 7 2 | 1 9 5 | 3 4 8
 1 9 8 | 3 4 2 | 5 6 7
-------+-------+------
 8 5 9 | 7 6 1 | 4 2 3
 4 2 6 | 8 5 3 | 7 9 1
 7 1 3 | 9 2 4 | 8 5 6
-------+-------+------
 9 6 1 | 5 3 7 | 2 8 4
 2 8 7 | 4 1 9 | 6 3 5
 3 4 5 | 2 8 6 | 1 7 9
```

17. Write a method that sums all the elements of a 1-dimensional array with an even index. Also write a method to sum all the elements with an odd index. Use these methods to see whether the even or odd indexed elements have a larger sum for an array of length 6 entered by the user.

Advanced - Adapt your solution to work for any length array, as decided by the user at runtime. As we have seen previously, the user should enter all values on 1 line, separated by spaces and terminated with a letter.

18. Write a method that inserts a new value into a sorted array. (Keep in mind that this will increase the size of the array). Assume that the array is sorted.

Advanced - Check online (or talk with a TA) about how to sort an array. Then, sort the array and apply the above method.

19. Implement a method that performs matrix addition and returns the sum of its two parameters as a result. Make the method check whether the sum is legal, i.e. the width and height of the matrices match, and warn the user if they don't. Write a main method to test it.

To make testing easy, you can instantiate, for example, a 3 by 3 matrix as follows:
double[][] matrix = {{1.0,2.0,3.0},{4.0,5.0,6.0},{7.0,8.0,9.0}};