CS 5000 – Summer 2023
Assignment #6, 50 Points

Single-Dimensional Arrays – Chapter 7

Develop a complete Java program for each of the following problems. Please name the programs as indicated and add proper program headers and output labels as shown below. ***Please use only concepts and programming constructs/syntax we discuss to date.***

Make sure you **include a header for each program (see previous assignments).**

Program #1 (25 points): Design and implement a Java program, named ***`CountOccurrences`***, as follows: the main method reads from the user, into a single-dimensional array of size 10 elements, **a number of positive integers between 1 and 100, up to 10 integers.** Then, it passes the array to another method (named `Count(…)`) to determine and printout the occurrence of each value in the array and printout the results as shown below. Assume that the user inputs end with 0 (sentinel value to stop the loop). The user may enter less than 10 numbers. Do not store the sentinel input value 0 into the array. The program should reject/ignore invalid inputs and continues to run.

Design the main method such that it allows the user to re-run the program with different sets of inputs (as in previous assignments, using a sentinel loop). Document your code and organize and space the outputs properly using escape characters as shown in the following sample outputs (notice `time` vs. `times`, refer to slide 33 and 34, Chapter 3).

The sample tests below show both the input prompt and the output labels. Please DO NOT read user inputs as String type. Notice that the user may enter one value per line (first test) or all values on one line (second and third tests). Please DO NOT sort the array as outputs will be different. Make sure your code displays the outputs following the test data format.

First test:

```
Enter up to 10 integers between 1 and 100 (0 to stop): 7
7
7
7
7
7
7
0

7 occurred  7 times
```

Second test:

```
Enter up to 10 integers between 1 and 100 (0 to stop): 5 2 3 2 1 -20
5 17 220 3 2 0

5 occurred  2 times
2 occurred  3 times
3 occurred  2 times
1 occurred  1 time
17 occurred 1 time
```

Third test:

```
Enter up to integers between 1 and 100 (0 to stop): 5 22 35 22 11 -35
55 356 45 35 11 0

5 occurred  1 time
22 occurred 2 times
35 occurred 2 times
11 occurred 2 times
55 occurred 1 time
```

```
45 occurred 1 time
```

Program #2 (25 points): Design and implement a Java method, named **`findIndex(…),`** in a separate class (and separate file) named ***`IndexOfLargest`***. The method takes a single-dimensional array of integer values as a parameter and returns the index of the largest value in the array. If the largest value appears more than once, the method returns the smallest index of that value. Notice that array indexing in Java starts with 0.

Next, write a test program in a separate file, named ***`TestIndexOfLargest`***, such that the main method asks the user to enter **10 integer values** into an array, then call method `findIndex(…)` in class `IndexOfLargest` to determine and display the index of the largest value in the array. Design the main method such that it allows the user to re-run the program with different sets of inputs (as in previous assignments, using a sentinel loop). Document your code and organize and space the outputs properly using escape characters as shown below. The sample test data below shows only the output labels. Please DO NOT read user inputs as String type. The sample tests below DO NOT show the input prompts, just the outputs. Again, the user may enter all numbers on one line one per line. Make sure your code displays the outputs following the test data format.

First test:
```
You entered these values:   15 23 -13 12 11 90 25 -17 90 90
Index of largest value:     5
```

Second test:
```
You entered these values:   5678 123 -113 12 211 -500 5678 17 51 100
Index of largest value:     0
```

Third test:
```
You entered these values:   -15 -3 -13 -62 -15 -51 -125 -117 -15 -10
Index of largest value:     1
```

Fourth test:
```
You entered these values:   125 -3 -13 -62 -11 541 -125 -117 555
1300
Index of largest value:     9
```

Submission:

1. Before submitting your programs, make sure you review the assignment submission requirements and grading guidelines posted in D2L. The grading guidelines explain some of the common errors found in programming assignments.
2. The assignment due date is posted in D2L.
3. Please compile and run your java files (only the .java files) right before you upload to the assignment submission folder in D2L.