New York University
Computer Science Department
Introduction to Software Development
Dr. Anasse Bari
Homework#5: ***Learning Arrays in Java***
Deadline: *See NYUclasses for the deadline, 15% off per day after the deadline (3 days maximum).*

Learning Objective:

- Learning and practicing arrays in Java
- Learning the difference between pass by reference and pass by value.
- Learning more about method overloading

Read the guidelines bellow carefully to avoid receiving a zero grade on the HW:

- Please use the following naming convention "ExerciseX.java", where X is the number of the exercise. Please apply this convention to all coming homework assignments during the semester. Also please attach the answers and include them into HW's zip file.
- Compile and run the program for each exercises.
- Comment Your Code (at least 5 comments per exercise)
- Make an archive (zip file or compressed file) with all the **java files (the .java files NOT the .class files)** and post it on NYU Classes (StudentName_Homework_X.zip)
- It is your responsibility to make sure that the Zip files has your actual files. You may send the file to yourself by email to double check that is the actual file before you upload on NYU classes.
- **If the graders cannot open the file, you will receive a grade of zero.**
- **If you send the .class files instead of the .java files (source files) you will receive a zero.**
- Please **do not send the entire project**, the Java file for each exercise is all that we need.
- An act of cheating will be severely addressed with an immediate zero on the homework and a report to the academic advisor and the administration.
- You will automatically lose 50%of the points for an exercise if the program does not compile and run correctly.
- The number of points for each questions is being specified.

- Plagiarized assignment will get a ZERO grade. You cannot change the variable names of other student's solution and submit it as yours. The program structure of other students must not match yours. Every student has to come up with his/her own solution. Any cheating (e.g. copying from internet without citing sources) is a serious violation of the University student code.

Guidelines:

- Compile and run the program for the input parameters specified for each exercises.
- Please use the following naming convention "*ExerciseX.java*", where X is the number of the exercise. Please apply this convention to all coming homework assignments during the semester.
- Make an archive (zip folder – Compressed Folder) with all the **java files** and post it on NYU Classes *(Homework_X.zip)*
- **Please comment you code. Explain major lines in your program.** Comments are should be brief and descriptive.
- Please **do not send the entire project**, the Java file for each exercise is all that we need.
- If you do not follow these guidelines, we will take points off.
- If you send .class files, you will get a zero.
- Late submissions: -5% on each day.

Exercises

Exercise#1 (25pts): (Reverse the numbers entered) Write a computer program that reads ten integers and display them in the reverse of the order in which they read.

Exercise#2 (25pts): Write a program that reads integers between 1 and 100 and counts the occurrences of each. Assume the input ends with 0.

Here is a sample run of the program:

```
Enter the integers between 1 and 100:  2 5 6 5 4 3 23 43 2 0
2 occurs 2 times
3 occurs 1 time
4 occurs 1 time
5 occurs 2 times
6 occurs 1 time
23 occurs 1 time
43 occurs 1 time
```

Exercise#3 (25pts): (Average an Array) Write two methods that return the average of an array with the following signatures:

```
public static double average (int [] array)
public static double average (double [] array)
```

Write a test program that prompts the user to enter 10 double values, invoke methods and displays the average value.

Exercise#4 (25pts): Implement the following method that returns true if the array is already sorted in increasing order.

```
public static boolean isSorted(int[] list)
```

Write a test program that prompts the user to enter a list and displays whether the list is sorted or not. Here is a sample run. Note that the first number in the input indicates the number of elements in the list. This number is NOT part of the list.

```
Enter list: 8 10 1 5 16 61 9 11 1
The list is not sorted

Enter list: 10 1 1 3 4 4 5 7 9 11 21
The list is already sorted
```