**EECS 233 Homework 9**

**General requirements:**

- Due at 11:00 PM on the posted due date.
- Include your name and network ID as a comment at the top of all of your programs.
- Create a typed document (.docx or .pdf) for the report with your name and network ID at the top.
- Upload your document and all .java files as a .zip file to Canvas. Do not use other formats such as .rar.
- All work should be your own, as explained in the Academic Integrity policy from the syllabus. File sharing is prohibited.

1. Revise each of the following programs, from the textbook, to sort greatest to least. Print the contents of the entire array, with indexes, as described. Also print the indexes and array *before* the sort. Exact output formatting in the examples, such as tabs, is not required. **Provide example output in your report.**

   a. Use Select.java. Print after every iteration of the outer loop of the sort. Below is an example:

```
Here is the entire original array:
[0]  [1]  [2]  [3]
80   10   50   70
Selection sort...
[0]  [1]  [2]  [3]
80   70   50   10
[0]  [1]  [2]  [3]
80   70   50   10
[0]  [1]  [2]  [3]
80   70   50   10
```

   b. Use Insert.java. Print after every iteration of the outer loop of the sort. Below is an example:

```
Here is the entire original array:
[0]  [1]  [2]  [3]
80   10   50   70
Insertion sort...
[0]  [1]  [2]  [3]
80   10   50   70
[0]  [1]  [2]  [3]
80   50   10   70
[0]  [1]  [2]  [3]
80   70   50   10
```

   c. Use Mergesort.java. Print twice: (1) before the two recursive calls dividing the array, and (2) after the halves are merged. For (1), indicate where the array is divided. Below is an example:

```
Here is the entire original array:
[0]  [1]  [2]  [3]
80   10   50   70
Dividing...
[0]  [1]   | [2]    [3]
80   10    | 50     70
Dividing...
[0]   | [1]
80    | 10
Merged...
```

```
[0]  [1]
80   10
Dividing...
[2]   | [3]
50    | 70
Merged...
[2]  [3]
70   50
Merged...
[0]  [1]  [2]  [3]
80   70   50   10
```

2. Complete the "partition" method in Quicksort.java, from the textbook, using the 1st array value as the pivot. **Provide example output in your report showing the initial array and the final array.** *WARNING: There are various implementations of mergesort available on the Internet. However, students are required to use only the techniques that have been presented in class. Additionally, students will be tested on their understanding of these particular techniques. Developing the program on your own will maximize your understanding.*

3. For each of the following algorithms, perform the given analysis on the array {10, 60, 40, 50, 30, 20} assuming it sorts greatest to least. **No programming is required** (though you are allowed to verify your answers using the programs described above)**.**

   a. Using selection sort, write the complete array after every iteration of the outer loop of the sort.
   b. Using insertion sort, write the complete array after every iteration of the outer loop of the sort.
   c. Using mergesort, draw a tree showing the divisions and merges of the subarrays, as demonstrated in class.
   d. Using quicksort, draw a tree showing each partitioned subarray, as demonstrated in class.

4. Give the average case big-O run times for each of the following algorithms. Explain in 1 – 3 sentences.

   a. Selection sort
   b. Insertion sort
   c. Mergesort
   d. Quicksort