CISC220 Homework 4

Due November 17th in lab (paper copy submission only).

Each question is marked as **[INDIVIDUAL]** and must be completed accordingly.

Academic Honesty Reminder

For parts that are marked as **[INDIVIDUAL]** you may discuss general algorithms with other people, but you may not share any of your code in **any** form. You may not help other people debug their programs, except in the very limited way described on the class website. You may not use any code written by others, whether they are students or not. The TA and the instructor are available for office hours, and by appointment if you can't make hours because of a class conflict.

Doing these exercises on your own will help prepare you for similar questions on the exam.

Assignment

For each question you will perform a sort on a specified series of input data. Assume that the representation for input data is an array of 32-bit integers. Your answer should fill in a "progress table" and/or a series of drawings showing the state of the algorithm at each step. You may not need all space or table rows provided.

Here is a reference for each algorithm (K+W is your textbook):

- Bubble Sort: K+W 577-580
- Selection Sort: K+W 572-576
- Insertion Sort: K+W 581-585
- (Self-Balancing) Binary Tree Sort: http://en.wikipedia.org/wiki/Binary_tree_sort
- Quick Sort: K+W 604-613
- Merge Sort: K+W 592-598
- MSD Radix Sort: http://en.wikipedia.org/wiki/Radix_sort

1. **[INDIVIDUAL] 10 pts.** Bubble sort.

   Example:

38	24	40	9	13	78	94	57	5	95
24	38	9	13	40	78	57	5	94	95
24	9	13	38	40	57	5	78	94	95
9	13	24	38	40	5	57	78	94	95
9	13	24	38	5	40	57	78	94	95
9	13	24	5	38	40	57	78	94	95
9	13	5	24	38	40	57	78	94	95
9	5	13	24	38	40	57	78	94	95
5	9	13	24	38	40	57	78	94	95

   Complete this bubble sort:

43	38	76	79	18	48	44	64	70	75

2. **[INDIVIDUAL] 10 pts.** Selection sort.

38	24	40	9	13	78	94	57	5	95
5	24	40	9	13	78	94	57	38	95

3. **[INDIVIDUAL] 10 pts.** Insertion sort. The * indicates the last element of the sorted partial result.

38*	24	40	9	13	78	94	57	5	95
24	38*	40	9	13	78	94	57	5	95

4. **[INDIVIDUAL] 10 pts.** (AVL balanced) Binary Tree Sort sort. Show the binary tree as the sort progresses. You do not have to show null data elements, they are shown in the first step for clarity.

38	24	40	9	13	78	94	57	5	95

1.
```
    38
   /  \
null null
```

2.

3.

4.

5.

6.

7.

8.

9.

10.

5. **[INDIVIDUAL] 10 pts.** Quick sort. The pivot should be chosen as the middle element of the array (not the mathematical median, just the element at floor(length / 2)). Place a * next to each element that is chosen as a pivot. Perform all recursive steps in parallel on each line.

   Example (only first 3 iterations):

38	24	40	9	13	78*	94	57	5	95
38	24	40	9*	13	57	5	78	94	95*
5	9	38	24	40*	13	57	78	94	95

   Complete this quick sort:

43	38	76	79	18	48*	44	64	70	75

6. **[INDIVIDUAL] 10 pts.** Merge sort. The merge group size starts at 2 and doubles each row; odd sized groups merge into the last group. Use asterisks as separators between merge groups. Example:

38 *	24 *	40 *	9 *	13 *	78 *	94 *	57 *	5 *	95
24	38 *	9	40 *	13	78 *	57	94 *	5	95
9	24	38	40 *	13	57	78	94 *	5	95
9	13	24	38	40	57	78	94 *	5	95
5	9	13	24	38	40	57	78	94	95

Complete this merge sort:

43 *	38 *	76 *	79 *	18 *	48 *	44 *	64 *	70 *	75

7. **[INDIVIDUAL] 10 pts.** MSD Radix sort. Show the partially sorted array after each step.

Step:	1	2	3	4	5
81472					
90579					
12698					
91337					
63235					
9754					
27849					
54688					
95750					
96488					

8. **[INDIVIDUAL] 30 pts.** Determine your personal preference ordering over the basic colors in the rainbow:

   Red, Orange, Yellow, Green, Blue, Indigo, and Violet

   For example, my personal preference ordering would be:

   Yellow $<$ Orange $<$ Red $<$ Violet $<$ Blue $<$ Green $<$ Indigo

   Your personal preference ordering is:

   _____ $<$ _____ $<$ _____ $<$ _____ $<$
   _____ $<$ _____ $<$ _____

   Using your personal preference ordering, perform an insertion sort on the following data:

Green	Orange	Yellow	Violet	Indigo	Red	Blue

   Now choose a different one of the algorithms from this assignment that will take fewer steps than insertion sort for your data. Show the execution of this algorithm for the same input data.