2) (10 pts) ALG (Sorting)

(a) (1 pt) Which of the sorting algorithms (listed in part d) could encounter problems if an array can contain duplicates? (Specifically, for four of the algorithms, whether or not there are duplicates in the array don't alter the run-time of the algorithm on individual cases, but one of the algorithms, in its original form, is definitively affected.)

Quick Sort (**Grading: 1 pt all or nothing)**

(b) (2 pts) What problem could be encountered?

If all of the numbers are the same, the partition element will either end up being the very first element or the very last element, always creating a recursive call on an array of size n – 1, if the previous call was on an array of size n. This results in an $O(n^2)$ run-time.

(c) (2 pts) Pick one of the algorithms that aren't affected by duplicates and explain why it runs similarly with or without duplicates.

Bubble: Regardless of what values are in the array, the sort always makes the same number of loop iterations, so duplicates don't affect the run-time at all.

Insertion: If we only continue our inner loop if the value being inserted is less than the value it's compared to, then equal values would not result in the inner loop running longer. Instead, a full array of equal values would take n steps to sort, since the inner loop would never repeat.

Merge: The merge algorithm does the same number of comparisons and copies regardless of the actual values in the two subarrays being merged. The Merge Sort always calls the Merge on arrays of the same sizes, so duplicates do not affect the run-time at all.

Selection: The loop structure of selection sort is similar to Bubble Sort and guaranteed not to change, regardless of the values, thus duplicates won't affect the run time of Selection Sort. (First time, loop runs through n values, then n – 1, then n – 2 and so forth.)

Grading: Give full credit if the choice of algorithm is clear and the reason is accurate. Give 1 point if you feel the answer is worth some credit but has some non-trivial ambiguities or holes.

(d) (5 pts) What is the worst case runtime for the following sorting algorithms on an array with ***n*** distinct values? Please list your answers with Big-Oh notation, using proper conventions.

Quick **$O(n^2)$**

Bubble **$O(n^2)$**

Insertion **$O(n^2)$**

Merge **$O(nlgn)$**

Selection **$O(n^2)$**

Grading: 1 pt per each one, don't give credit if there are leading constants