Pre-lab Part 1

1. How many rounds of swapping do you think you will need to sort the numbers 8,22,7,9,31,5,13 in ascending order using Bubble Sort?
2. How many comparisons can we expect to see in the worse case scenario for Bubble Sort? Hint: make a list of numbers and attempt to sort them using Bubble Sort.

1.9 rounds
2.O(n^2)

Pre-lab Part 2

1. The worst time complexity for Shell sort depends on the size of the gap. Investigate why this is the case. How can you improve the time complexity of this sort by changing the gap size? Cite any sources you used.
2. How would you improve the runtime of this sort without changing the gapp size?

1.The idea of Shell sort is to rearrange the array using a sequence of values of h that ends in 1, and then take every h th entry yields a sorted subsequence. The efficiency of shellsort is decided by making a tradeoff between size and partial order in the subsequences, which says it's decided by the size of the gap. To improve the time complexity, we can dynamically adjust the size of the gap based on the length of the array to be sorted.
2.Chose an increment sequence whose performance on randomly ordered arrays has not been precisely characterized. Then the runtime of this unchanged gap size will be improved averagely.
source: Algorithms, 4th edition(Sedgewick, 1983)

Pre-lab Part 3

1. Quicksort, with a worse case time complexity of $O(n^2)$, doesn't seem to live up to its name. Investigate and explain why Quicksort isn't doomed by its worst case scenario. Make sure to cite any sources you use.

1.

- Moves: The inner loop of quicksort increments an index and compares an array entry against a fixed value. Quicksort achieves a better simplicity because it does not have data movement inside the inner loops.
- Compares: Quicksort uses few compares.
- The best case for quicksort is when each partitioning stage divides the array exactly in half. The average number of compares in quicksort is only about 39 percent higher than in the best case.

source: Algorithms, 4th edition(Sedgewick, 1983)

Pre-lab Part 4

1. Can you figure out what effect the binary search algorithm has on the complexity when it is combined with the insertion sort algorithm?

1.The worst case for insertion sort algorithm has a quadratic running time O(n^2). The binary search algorithm performs logn running time when determining the correct location to insert new elements, which changes the worst case for insertion sort algorithm into O(nlogn).
source:https://en.wikipedia.org/wiki/Insertion_sort