ECE368 Project 1 Report
Shijin Wang

Description of the algorithms:

In project 1, we were asked to write two algorithms, Shell_Insertion_Sort and Improved_Bubble_Sort.

For Shell sort, we first generate sequence 1 as k values by using two , then I use the K values as an array to perform the inseration sort by sorting each array in K gaps and decrement K by sequence 1.

For improved bubble sort, We first generate sequence 2 as K values, then I also use an K array to count the bubbles from the larger value to the smaller value to reduce extra run time .

Analysis of the time- and space- complexity:

For sequence1, the time complexity is O(n), space complexity is O(1).

For Sequence2, the time complexity is O(log(n)), space complexity is O(1).

	1000.txt	10000.txt	1000000.txt
Improved_Bubble(comparsion)	23408	342659	55067886
(Move)	13095	187818	30237996
(Run_time/s)	0	0	0.2
Shell_Insertion(comparsion)	30955	550711	123987151
(Move)	66221	1166240	259684562
(Run_time/s)	0	0	0.5

Run time: The run time of shell insertion sort is about twice as much as the Improved Bubble sort.

Number of comparisons: The comparsions of insertion sort are about twice as many as bubble sort, however, as the input grows, the comparsions of insertion sort grows more.

Number of Moves: The moves of insertion sort are about twice as many as bubble sort, however, as input grows, the moves of insertion sort grows more.

Summary:

The space complexity of both sorting algorithms are O(1).