Justin-Anthony Joco 02/07/2017
ECE 368 PA02 Report ID: 0028483667

In this assignment, the goal was to conduct shell insertion sort on a binary file through linked lists, instead of arrays. Memory and runtime data is recorded below:

Figure 1: Algorithm Output Information

Number of Integers	**Shell Insertion Sort with Arrays**			**Insertion Sort with Lists**		
	I/O Runtime (s)	**Sorting Runtime (s)**	**Memory Allocated (bytes)**	**I/O Runtime (s)**	**Sorting Runtime (s)**	**Memory Allocated (bytes)**
15	0	0	1972	0	0	1504
1,000	0	0	17,732	0	0	17,776
10,000	0	0	161,732	.49	.37	162,208
100,000	.02	.02	1,601,732	101	91	1,602,752
1,000,000	.36	.35	16,001,732	Long time	Long time	~16 million

Unfortunately, I was unable to implement shell sort using linked lists. Nonetheless, I was able to conduct insertion sort on the entire list. As a result, the I/O and sorting runtimes of the linked lists was much higher than that of the array's. Theoretically, linked list shell insertion sort could be as fast as array shell sort. It is noted, however, that lists allocate more memory due to the use of element pointers.

According to the table, the memory complexity of the linked list sort is O(n), similar to that of the array. Unlike the array shell sort, whose runtime complexity is O(logn), the runtime complexity of the list is about O(2^n), which explains why the runtime increases exponentially as order of magnitude of the integer number increases.

Had I implemented Shellsort of the linked list, both the time and memory complexities could be similar. In a real-world application, linked lists would be cheaper than arrays to store and delete data due to how dynamic and flexible they are. Unfortunately, one cannot access a specific element index within the list without traversing through the list; on the other hand, one can access a specific element index relatively easily. In addition, due to how easily accessible elements are within arrays, programs using arrays can perform better than those using linked lists. Thus, one must consider time and money as factors of choosing to implement a program using arrays or linked lists.