Pranjit Kumar Kalita
Project #1

1. **For Generating Sequencing for shell Sort**:
(a) Time Complexity: power() function – O(n)
largest_index() function – O(log n)
array_create() function – O(n^2)
Thus, the time complexity for generating sequence will be O(n^2).

(b) Space Complexity: power() function – O(1)
largest_index() function – O(1)
array_create() function – O(n) (for creating the memory allocation for array that stores sequence.
Thus, the space complexity for generating sequence will be O(n).

2.

Shell Sort using Insertion Sort					
File Name	# of Comparisons	# of Moves	I/O Time	Sorting Time	Comments
100000.txt	1.462139e+07	7.310693e+06	0.000000e+00	0.000000e+00	
1000000.txt	2.026664e+08	1.013332e+08	0.000000e+00	0.000000e+00	

Shell Sort using Selection Sort				
File Name	# of Comparisons	# of Moves	I/O Time	Sorting Time
100000.txt	1.499039e+10	2.811402e+06	0.000000e+00	6.200000e+01
1000000.txt	No output	No output	No output	No output

Comments: For Shell Sort using Insertion Sort, the Time Complexity was determined to be O(n.log n). For Shell Sort using Selection Sort, the Time Complexity was determined to be O(n^2).
Thus, selection sort takes more time than insertion sort, and the corresponding increase in the no. Of comparisons and moves are much more in selection sort than in insertion sort. In fact, it is the result of this exponential increase that the selection sort does not display any output for the 1 million data items case, whereas insertion sort works just fine.

3. **Space Complexity for Shell Sort using Insertion Sort**:
- allocate memory for sequence – O(n)
- variables – O(1)
- largest_index() function call – O(log n)
Inside the loops, no further memory is being allocated. Thus, space complexity is O(n).
Space Complexity for Shell Sort using Selection Sort:
- allocate memory for sequence – O(n)
- variables – O(1)
- largest_index() function call – O(log n)
Inside the loops, no further memory is being allocated. Thus, space complexity is O(n).