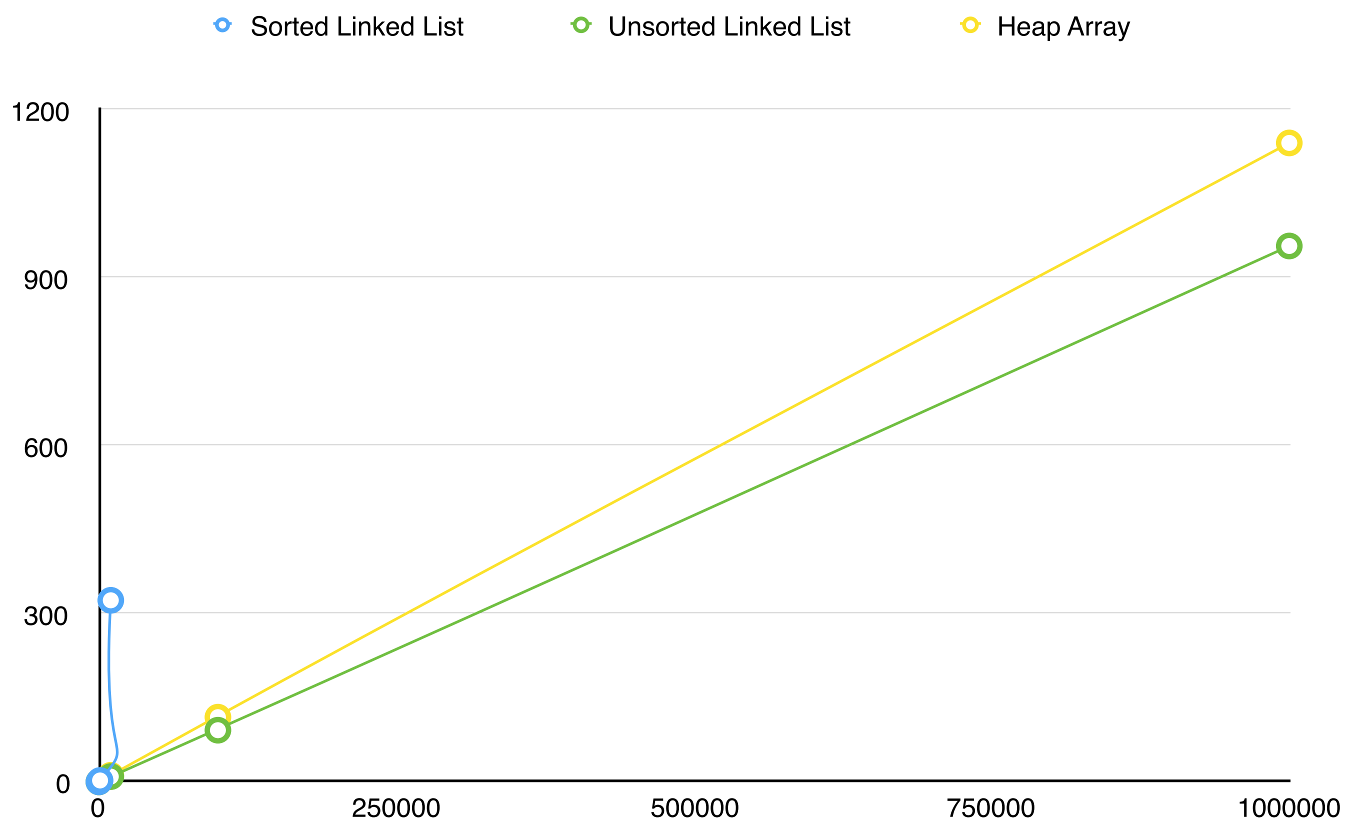
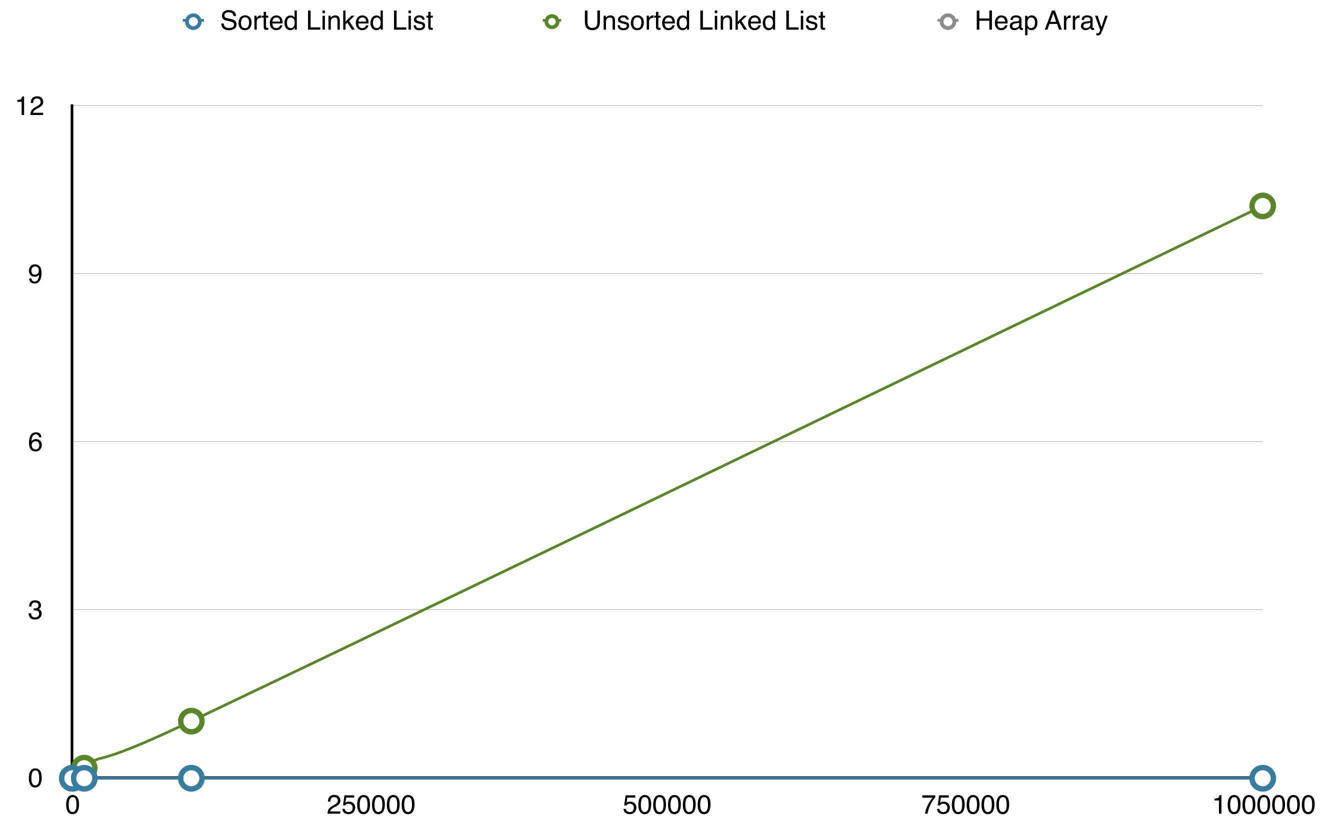
Enqueue Performance Plot



Dequeue Performance Plot



Test Procedure

A function named “analyze(T& queue)” was added to the provided test harness. “analyze(T& queue)” first enqueued strings on multiples of ten. For the enqueue, I tested from 1, 10, 100 … 10,000 strings on the **sorted** implementation. I could not go any higher since as evidence from the graph, the average elapsed times began increasing exponentially and I couldn't get a time for 100,000 strings. With the **unsorted** and **heap** array implementations, the running times were more manageable, and for both I went up to 10,000,000 strings. I then tested each multiple of 10 on 50 repetitions to average out a final elapsed time to plot on the graphs. For dequeue, I simply used the same newly filled queues of multiple of 10 strings and averaged out the time for 1 dequeue operation.

Conclusions

The sorted linked list implementation starts taking unmanageable amounts of time as the number of strings increases to enqueue. From 1,000 to 10,000 strings, the time jumped from 2.56 to 324.58. I could not even plot values higher than 10,000 strings. Big-O complexity for sorted linked list is quadratic—O(N^2). I did not expect the enqueue operations for the heap and unsorted implementations to be neck-and-neck. Unsorted is expectedly slightly faster, but the bubble operations used in the heap array add a surprisingly little amount of time to elapsed times. Big-O of both is linear—O(N). Had it not been for dequeue times, it would be clear which implementation is obviously better. The sorted and heap array implementations both run in constant time—O(1), while the unsorted implementation grows at linearly at O(N).

It is clear that the sorted linked list implementation becomes unusable at large enqueue operations. This leaves unsorted and heap. The unsorted linked list implementation is slightly faster at enqueueing than heap, but ultimately this does not make up for the linear dequeue time against the constant time dequeue of the heap implementation.

Big-O Chart

	Sorted Linked List	**Unsorted Linked List**	**Heap Array**
Enqueue	O(N^2)	O(N)	O(N)
Dequeue	O(1)	O(N)	O(1)