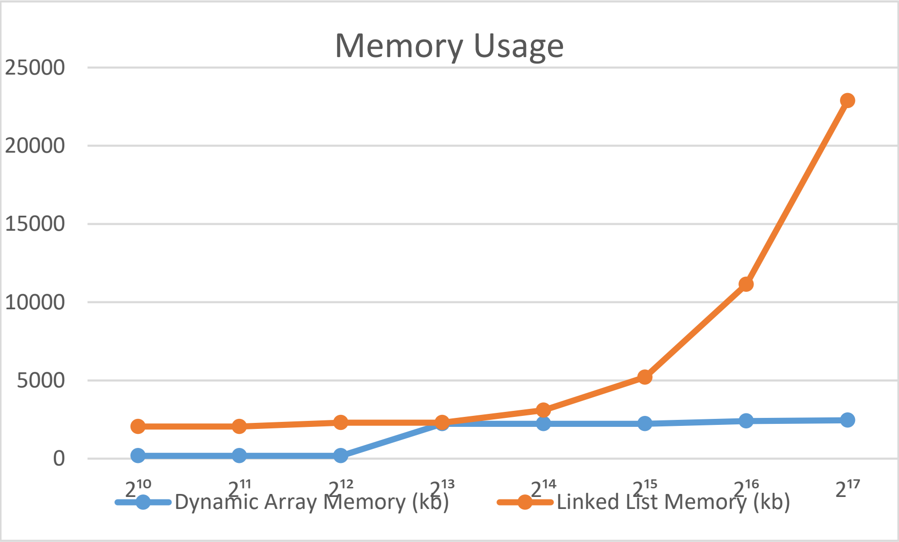
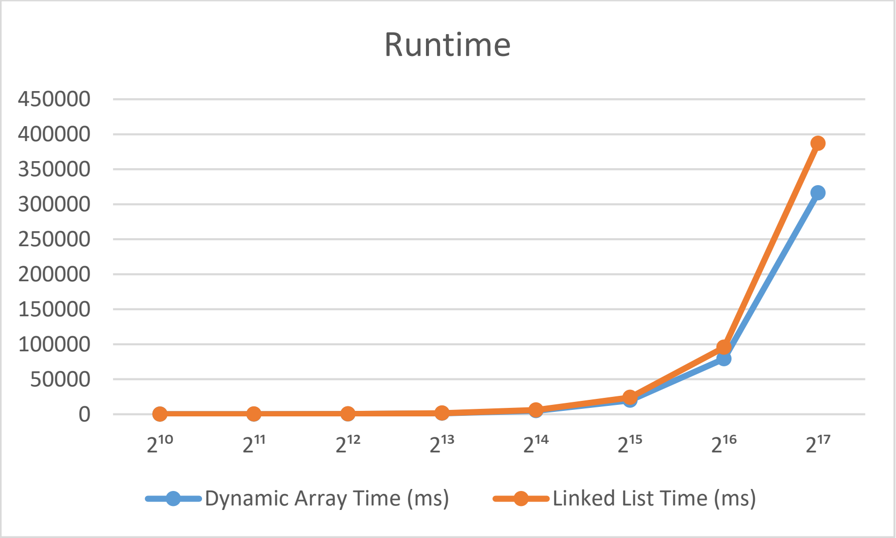
Dynamic Array vs Linked List – Memory & Runtime

Table:

Elements	Dynamic Array: Memory Used (kb)	Linked List: Memory Used (kb)	Dynamic Array: Runtime (ms)	Linked List: Runtime (ms)
n = 2^{10} = 1024	180	2048	30	20
n = 2^{11} = 2048	180	2048	80	100
n = 2^{12} = 4096	180	2300	310	380
n = 2^{13} = 8192	2228	2300	1260	1510
n = 2^{14} = 16384	2228	3092	4990	6000
n = 2^{15} = 32768	2228	5204	19870	23990
n = 2^{16} = 65536	2404	11140	79220	95580
n = 2^{17} = 131072	2452	22884	316160	386950

Plots:





Questions:

1. The Linked List implementation uses more memory. This is because each element in the linked list requires memory allocation for an integer and 2 pointers. An element in the dynamic list only requires allocation for an integer. As the number of elements increases exponentially, so does the size of the linked list.
2. The Dynamic Array implementation is fastest. This is because the computer can just jump from one element in memory to the next instead of having to use pointers to navigate to each element. However, the difference in runtime is not great, as both increase at a similar rate relative to the number of elements.
3. I would expect the runtime of the dynamic array implementation to increase greatly because all elements after the one being removed have to be moved up one element in the array. Doing this thousands of times will inevitably take a while. Removing an element from a linked list takes less shuffling around so the runtime won't increase as sharply.