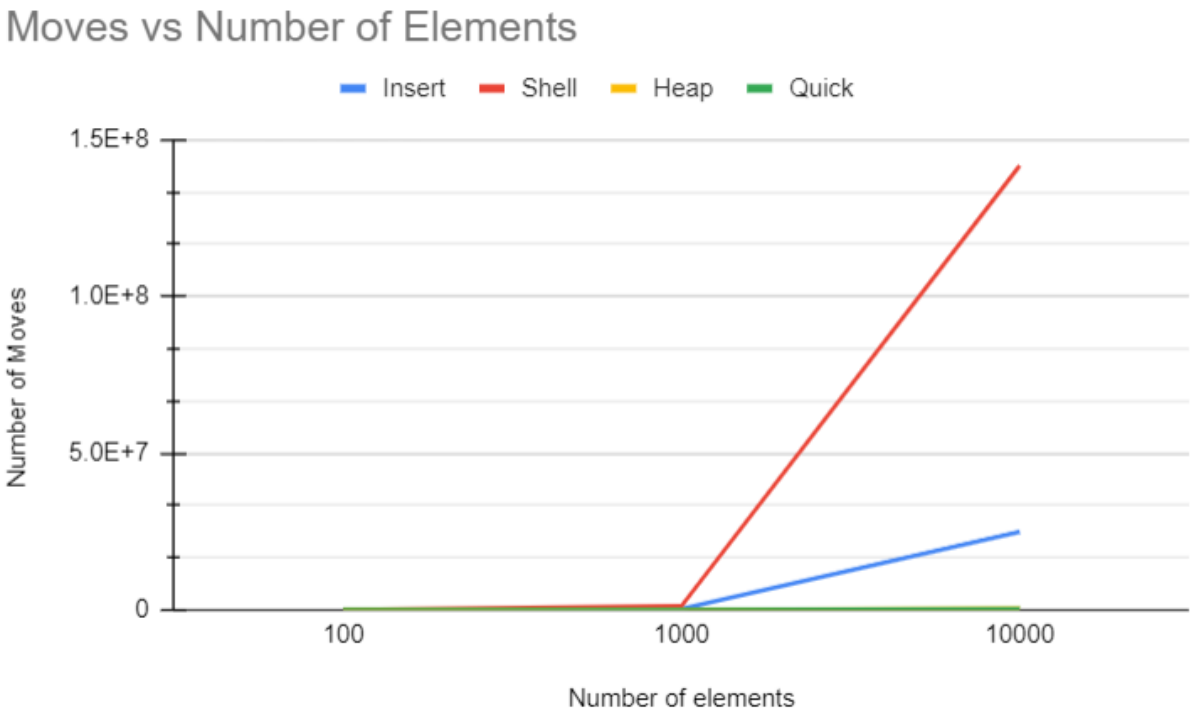
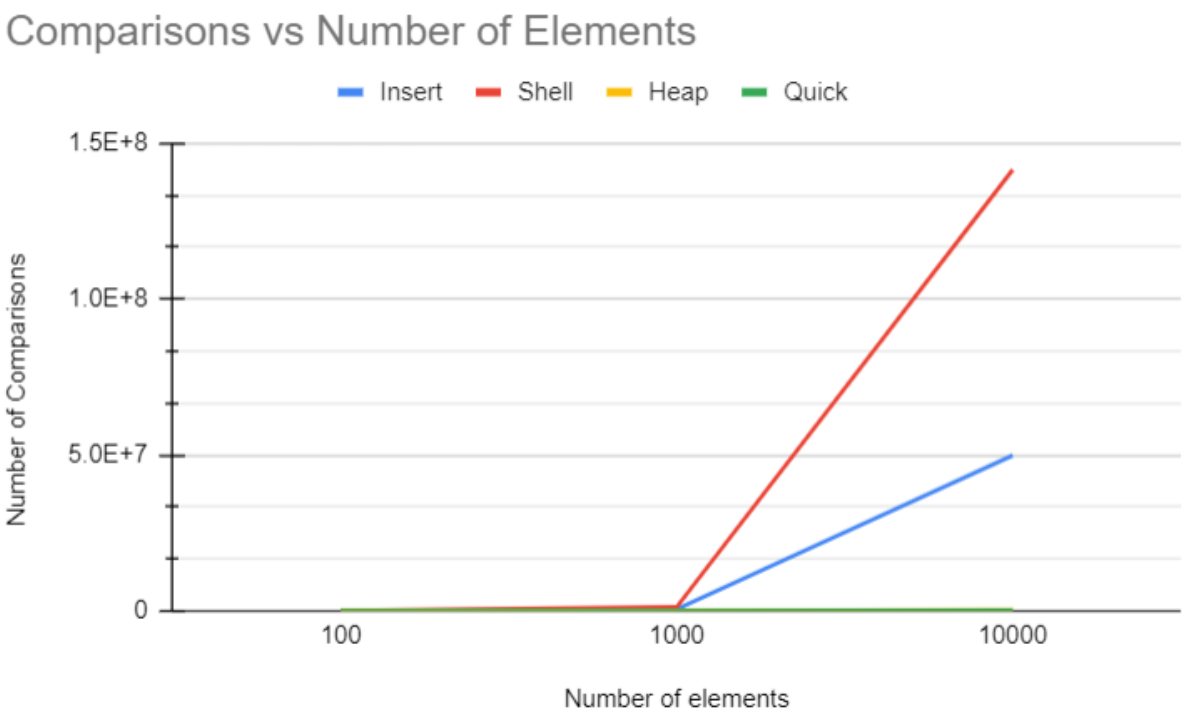
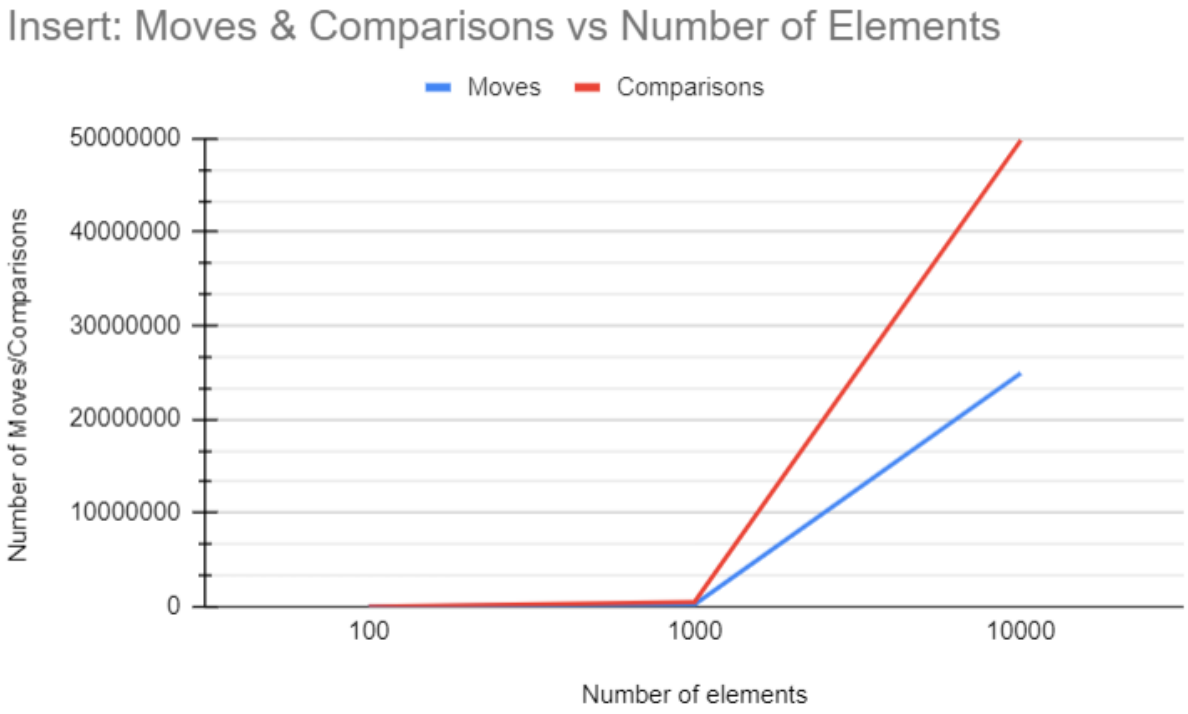
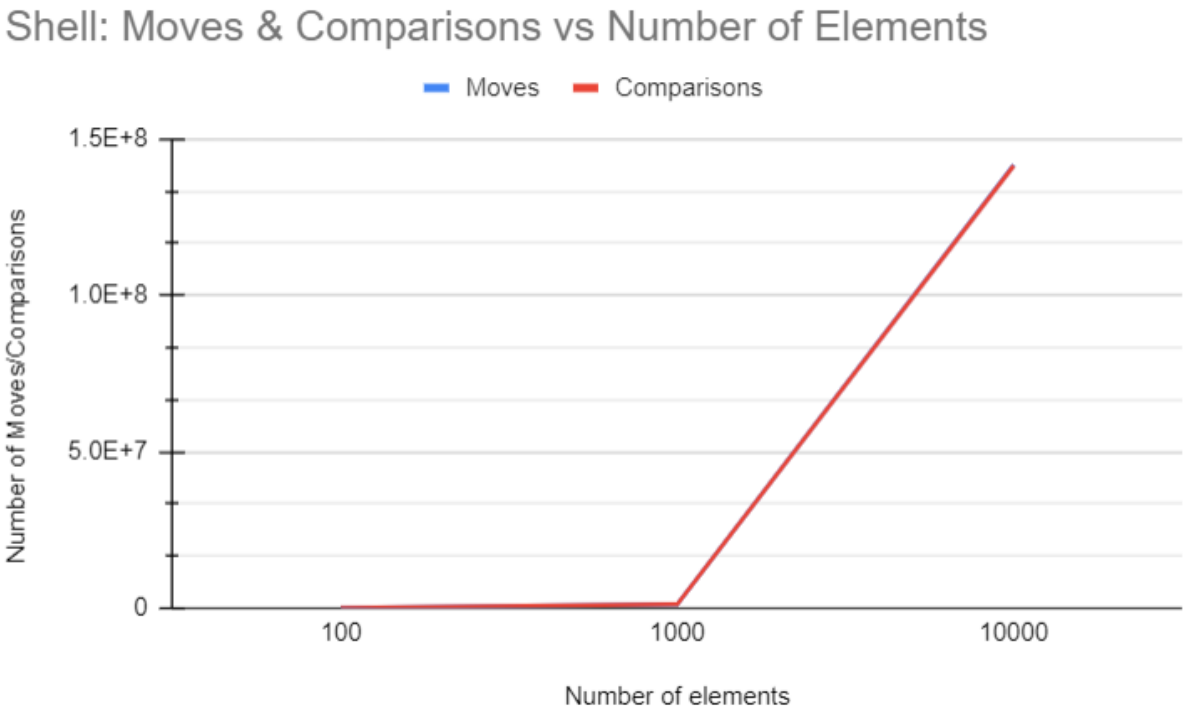


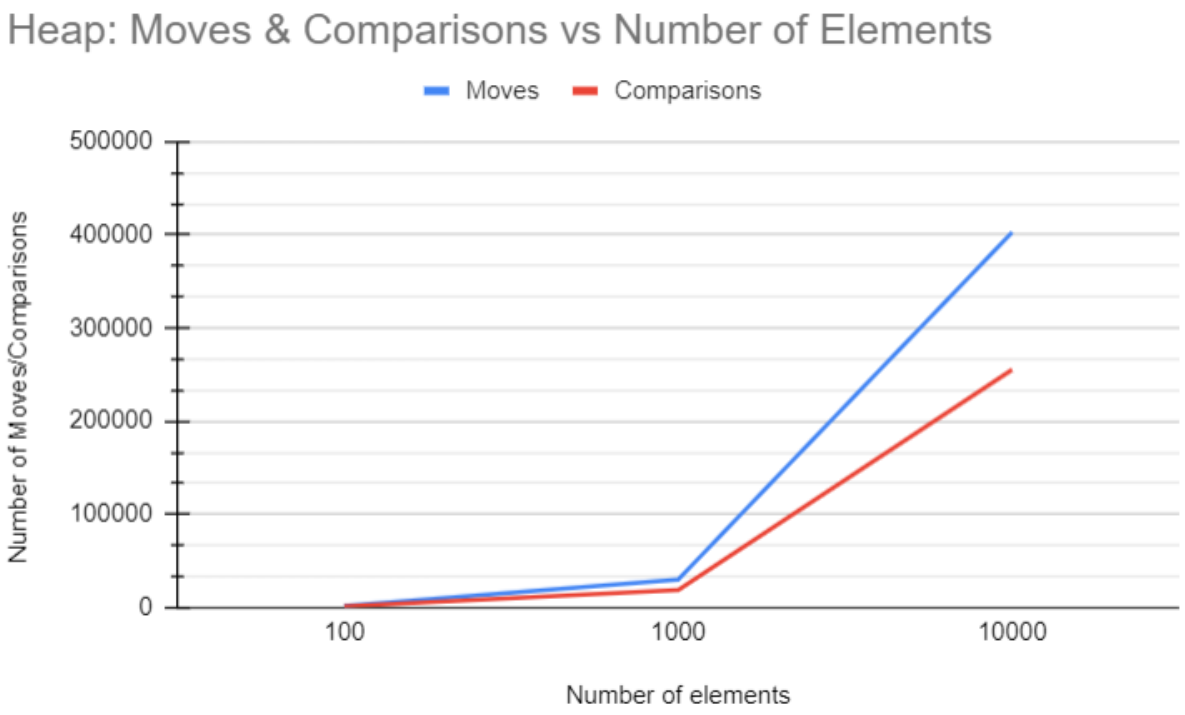


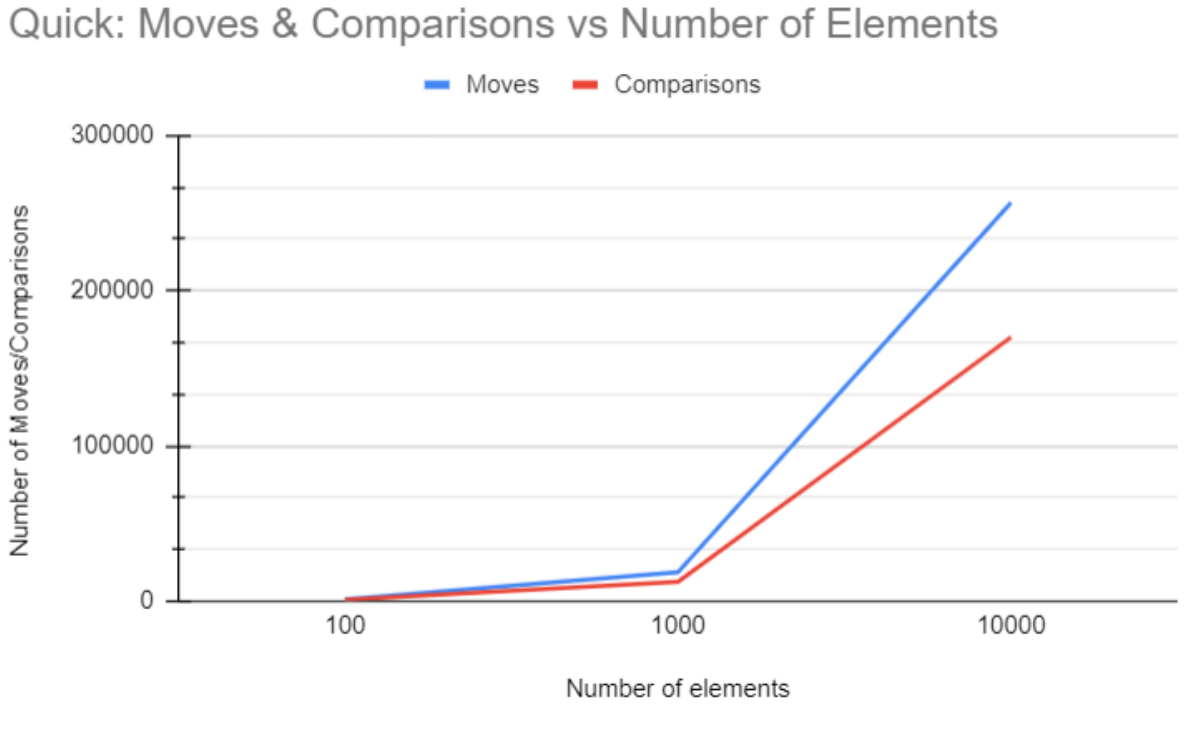
From the graphs above we can conclude that below 1000 elements, the sorting algorithm does not matter significantly. We can see that quick sort and heap sort perform well even with very high numbers. This is because the partitions used in quick sort prevent it from reevaluating and comparing large segments of the array that it has already looked at. Similarly with heap sort,

organizing the array so that small numbers are filtered to the bottom of the heap prevents the program from having to relook at already sorted sections of the array excessively. This is why with both heap and quick sort that we see fewer moves and comparisons with high numbers of elements.









By looking at each sorting algorithm individually we can see that the more efficient algorithms, heap and sort have a higher number of comparisons compared to moves. Similarly to above we also see the sharp change at 100 elements when all algorithms start using considerably more moves and comparisons.