• Identify the respective time complexity for each sort and include what you have to say about the constant

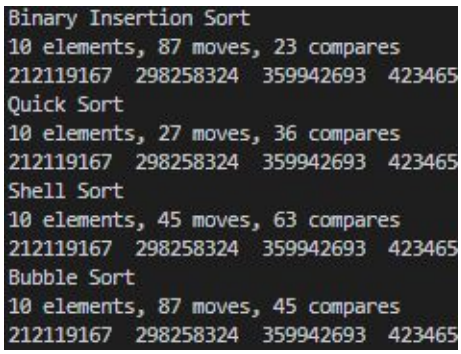
Bubble sort had an n^2 time complexity. QuickSort had around a n Log(n) time complexity which is quite a bit faster than n^2. Shell sort also had a time complexity around n log(n), binary insertion sort also has a time complexity of n log(n). The constant n number of elements had a large impact on the amount of time each sort took and as it increased in my testing, it took much longer to compute given the time complexities of each algorithm. The larger n was the more of a lead quick sort seemed to take in terms of moves and compares as shown in the examples below and on the next page

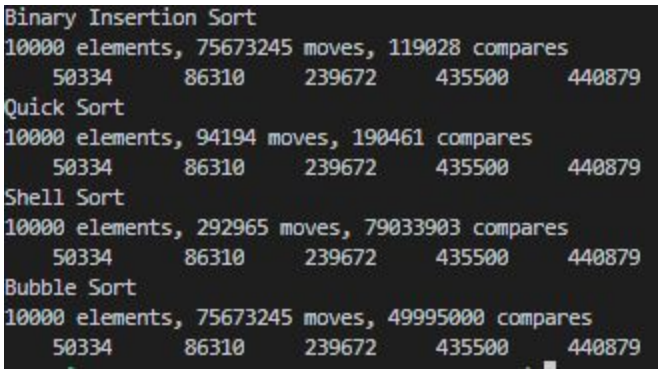
.• What you learned from the different sorting algorithms.

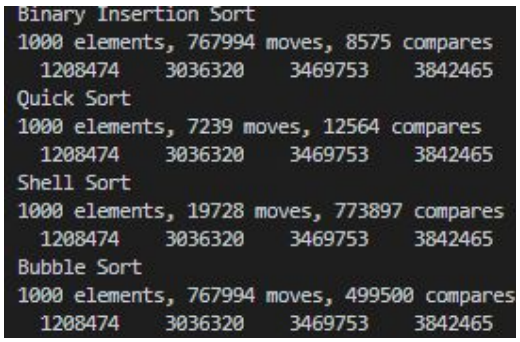
These algorithms have taught me different ways to look at solving the same fundamental problem. While bubble sort results in the same output as every other algorithm, it does it quite slower. By thinking about the problems bubble sort faces, the other algorithms work to attack these problems and create a faster overall algorithm. By looking at the problems facing a program, it's possible to improve the program and create a better, faster one. By looking at the moves and compares I learned a lot about how the different algorithms work to sort the list as some choose to compare more to sort while others choose to move and compare a similar amount of times.

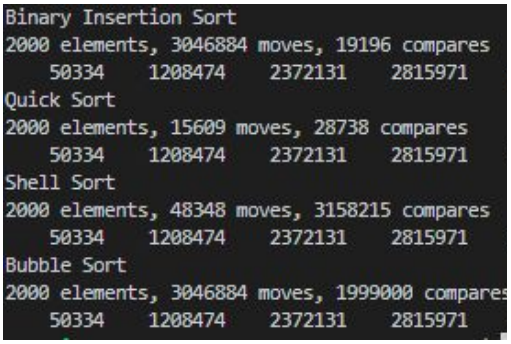
• How you experimented with the sorts.

I experimented with the different types of sorting algorithms by testing with a different number of elements in the arrays as shown in the examples below and on the next page. By doing this I was able to gain a better understanding of how an increasing number of elements affects the different algorithms. For example in larger arrays, quick sort seemed to be the quickest sorting algorithm as it had less moves and compares than the other ones. However in smaller arrays almost every sorting algorithm was comparable to one another so the easiest one to implement (bubble sort) would be the one I would use in those cases.






Binary Insertion Sort
1000 elements, 767994 moves, 8575 compares
1208474 3036320 3469753 3842465
Quick Sort
1000 elements, 7239 moves, 12564 compares
1208474 3036320 3469753 3842465
Shell Sort
1000 elements, 19728 moves, 773897 compares
1208474 3036320 3469753 3842465
Bubble Sort
1000 elements, 767994 moves, 499500 compares
1208474 3036320 3469753 3842465


Binary Insertion Sort
2000 elements, 3046884 moves, 19196 compares
50334 1208474 2372131 2815971
Quick Sort
2000 elements, 15609 moves, 28738 compares
50334 1208474 2372131 2815971
Shell Sort
2000 elements, 48348 moves, 3158215 compares
50334 1208474 2372131 2815971
Bubble Sort
2000 elements, 3046884 moves, 1999000 compares
50334 1208474 2372131 2815971