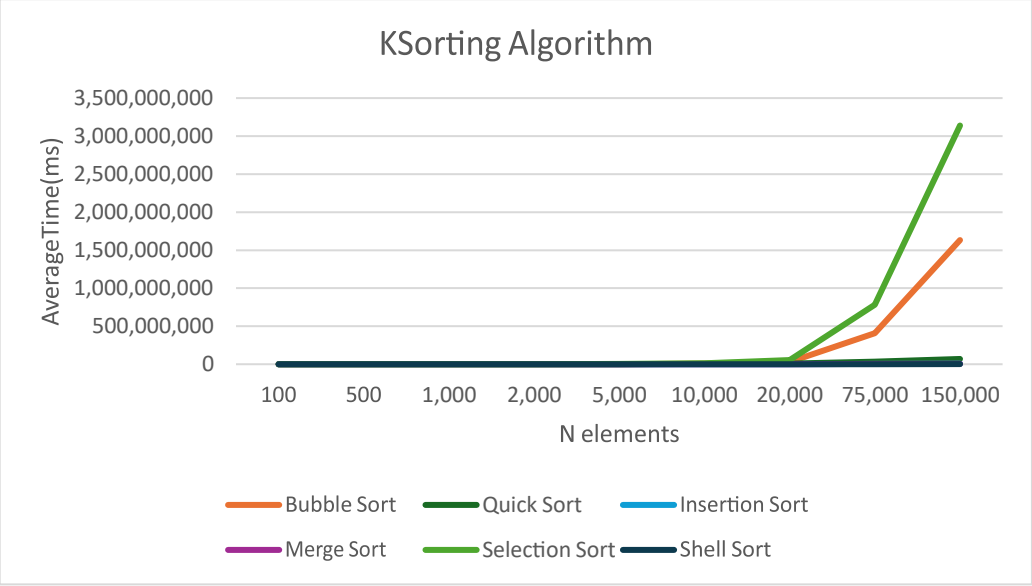
Question 12

N elements we are sorting	Bubble Sort	Quick Sort	Insertion Sort	Merge Sort	Selection Sort	Shell Sort
100	70,125	11,700	15,295	23,410	53,840	20,290
500	176,525	32,260	59,290	57,535	189,340	85,340
1,000	91,415	33,035	50,885	88,410	153,015	59,300
2,000	392,000	266,470	22,240	109,780	579,030	25,280
5,000	1,844,815	859,815	70,945	355,690	3,810,540	89,705
10,000	7,729,580	2,711,020	138,690	531,620	14,110,615	168,350
20,000	29,724,400	7,541,435	456,690	863,875	55,965,415	387,825
75,000	408,524,180	32,933,860	2,373,440	4,934,075	781,486,925	983,535
150,000	1,632,681,770	69,086,415	6,723,145	5,560,190	3,139,402,230	1,802,105



After doing the KSorted data I noticed that everything performed pretty similarly to my first ranking for the random data but did have some differences due to some sorting algorithms performing way better with more elements given. For example, Shell sort performed the best with the 150,000 values given, but in the random data it was one of the worst. Some of these algorithms such as shell sort performed better with more elements, because they were partially sorted and algorithms such as shell tend to do better with partially sorted arrays. Also, just as in problem 9 it was hard to fully represent the graph due to having values that vary so far from each other, which is why it is hard to graph numbers that are so big.