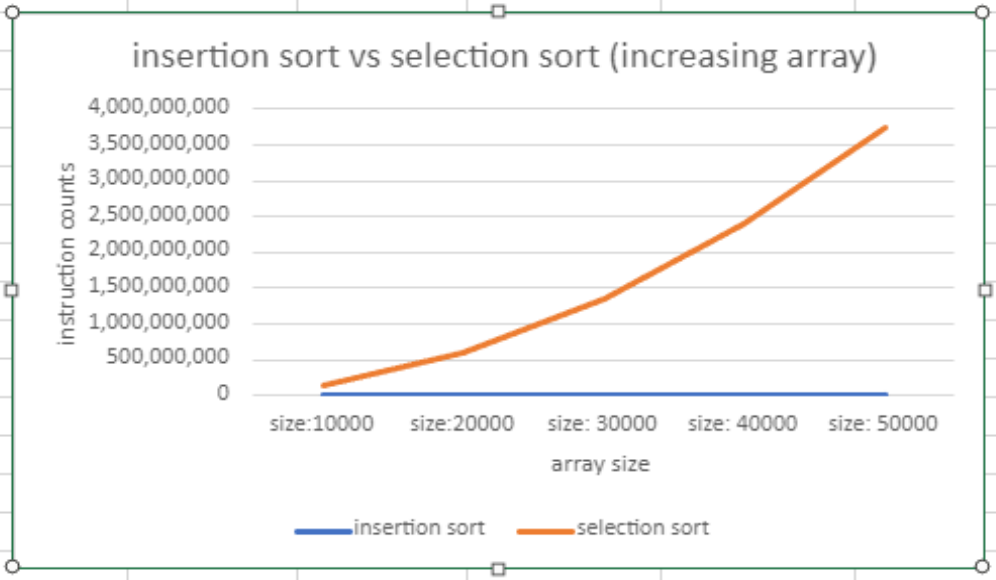
	A	B	C	D	E	F
1		size:10000	size:20000	size: 30000	size: 40000	size: 50000
2	insertion sort	59,996	119,996	179,996	239,996	299,996
3	selection sort	150,054,995	600,109,995	1,350,164,995	2,400,219,995	3,750,274,995



The graph and data show an astronomical difference in efficiency between the two sorting algorithms. While the selection sort algorithm has a similar instruction count with the other two possible orientations and numbers into the billions, the insertion sort algorithm's instruction count never even breaks a million. The graph of the selection sort algorithm curves upwards, while the insertion sort's graph appears to be a straight line. An initially increasing array orientation is the best possible case for both algorithms, but the insertion sort algorithm is able to capitalize on it to a far greater degree than selection sort. When compared to the other 2 initial data orientations, one may notice that the selection sort's instruction counts are very similar to this graph despite the increasing orientation being the best possible case. This is due to the nature of the selection sort algorithm's inner loop, which checks every element in the unsorted portion of the array per iteration of the inner loop. This will happen regardless of the array's initial orientation and thus the selection sort algorithm cannot take advantage of the best case as effectively. The insertion sort algorithm's inner loop, on the other hand, only runs when the values to a chosen cell's left are greater than the chosen cell's value, at which point it will move the greater value to the right. Because the array is initially in increasing order, the lines in the inner loop never run and allows the insertion sort algorithm to leave the selection sort in the dust.