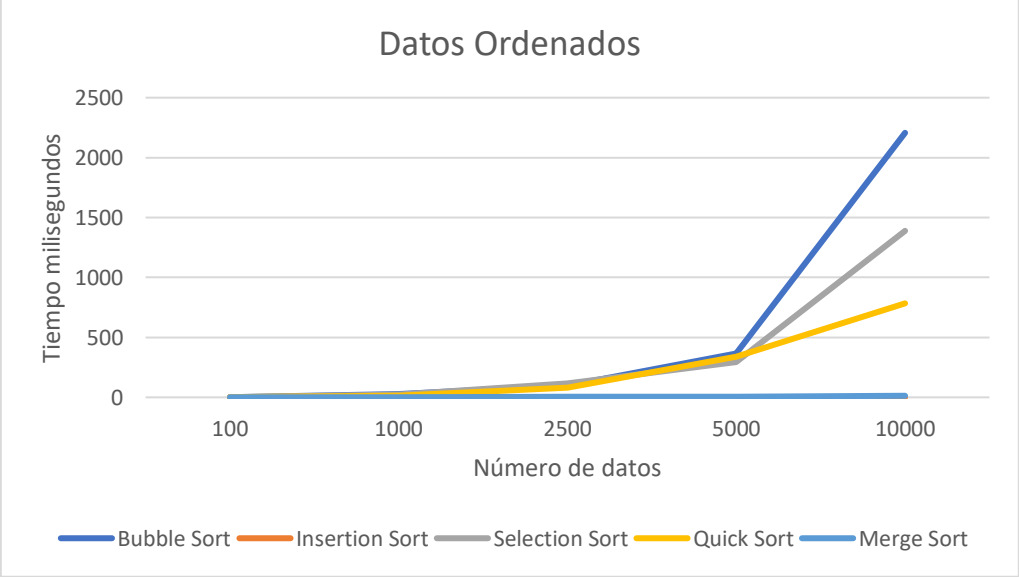
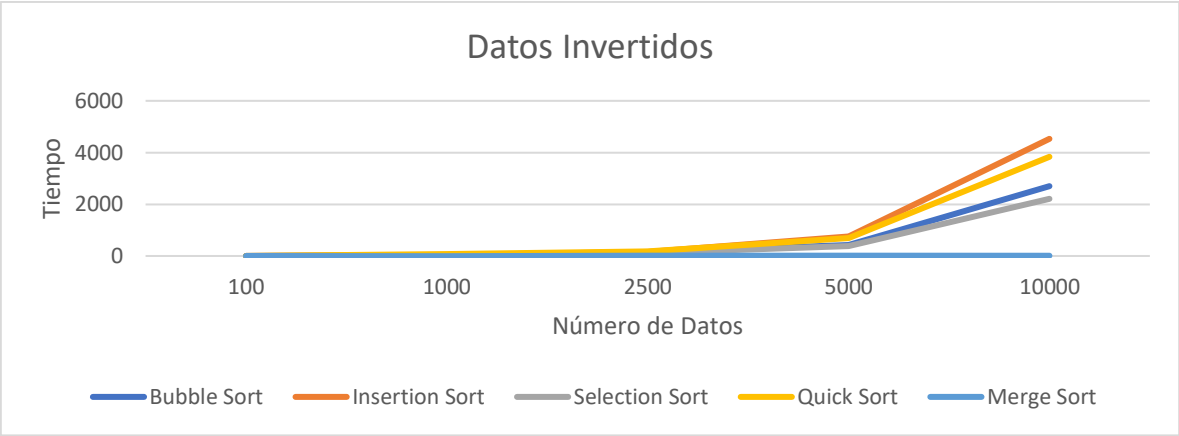
Fonseca Zárate Israel 183708

TIEMPO (MILISEGUNDOS)	DATOS ORDENADOS				
	100	1000	2500	5000	10000
BUBBLE SORT	1.01598	25.3643	98	363.51	2206.4292
INSERTION SORT	0.22542	0.2015	0.169499	0.695799	1.0234
SELECTION SORT	0.994666	21.9134	114.166399	293.848399	1388.7776
QUICK SORT	1.547066	13.284533	82.1745	340.079399	783.96435
MERGE SORT	0.551733	1.807333	2.8332	5.077	12.9535



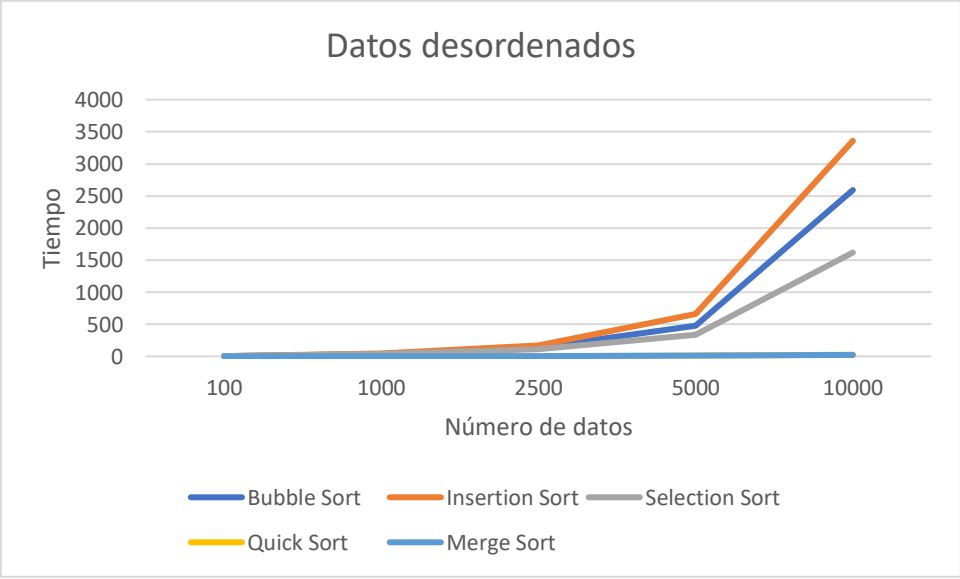
Como se ve en la gráfica, con datos ordenados, resulta más efectivo los métodos de merge sort e insertion sort; por otro lado, el resto de métodos cuantos más datos reciben, el tiempo aumenta mucho más rápido.

TIEMPO (MILISEGUNDOS)	DATOS INVERTIDOS				
	100	1000	2500	5000	10000
BUBBLE SORT	1.6114	30.8127	133.4501	418.3079	2699.7707
INSERTION SORT	1.4748	39.2216	158.1489	764.1786	4530.8572
SELECTION SORT	0.8235	21.3406	117.7518	388.6036	2209.7766
QUICK SORT	1.4316	77.3434	180.8016	701.713	3835.5223
MERGE SORT	0.2111	2.0587	3.0704	5.5719	9.3664



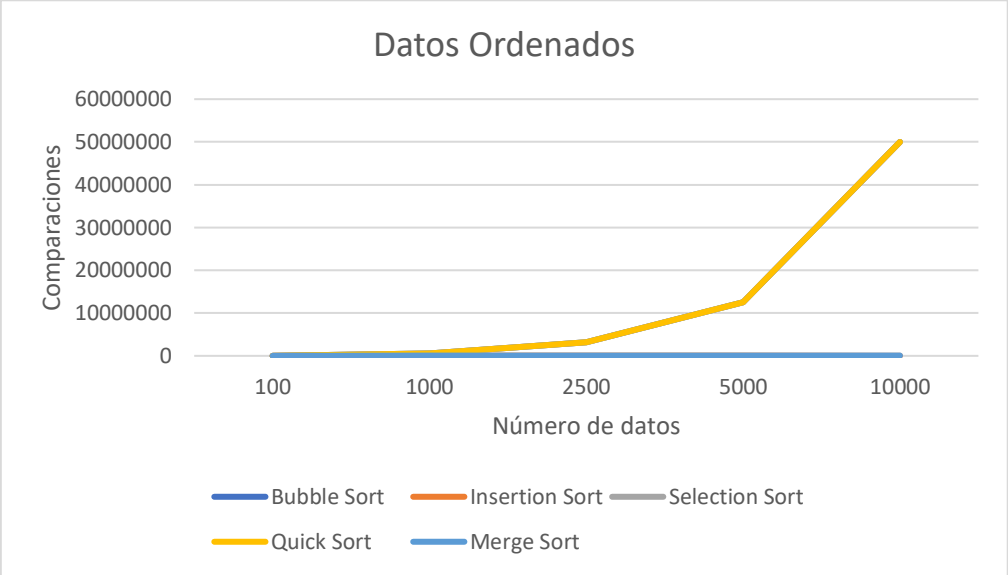
Con datos invertidos, el algoritmo de Merge Sort sigue siendo el más eficaz, por otro lado, los otros métodos, poseen tiempos similares.

TIEMPO (MILISEGUNDOS)	**DATOS DESORDENADOS**				
	100	1000	2500	5000	10000
BUBBLE SORT	3.8197	36.7246	138.8259	473.7683	2592.2592
INSERTION SORT	3.9867	49.6085	166.4687	661.9335	3359.4987
SELECTION SORT	3.7354	37.323	114.2055	339.3306	1618.2119
QUICK SORT	2.9454	3.7524	6.25	12.3817	24.8739
MERGE SORT	2.8687	4.3569	6.8147	11.8238	23.3534



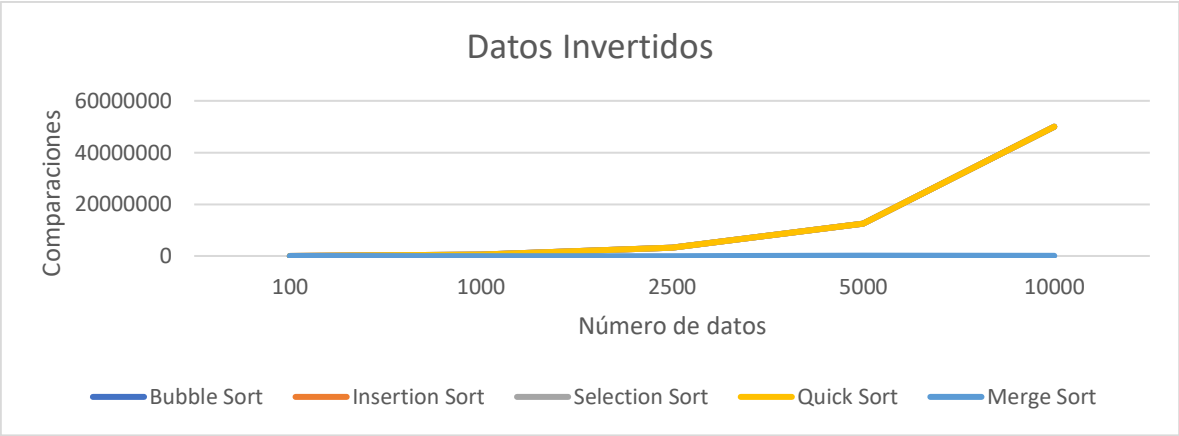
Con los datos desordenados, merge sort y quick sort son los más eficaces, insertion sort es el algoritmo más ineficaz.

COMPARACIONES	DATOS ORDENADOS				
	100	1000	2500	5000	10000
BUBBLE SORT	4950	499500	3123750	12497500	49995000
INSERTION SORT	99	999	2499	4999	9999
SELECTION SORT	4950	499500	3123750	12497500	49995000
QUICK SORT	4950	499500	3123750	12497500	49995000
MERGE SORT	356	5044	14752	32004	69008



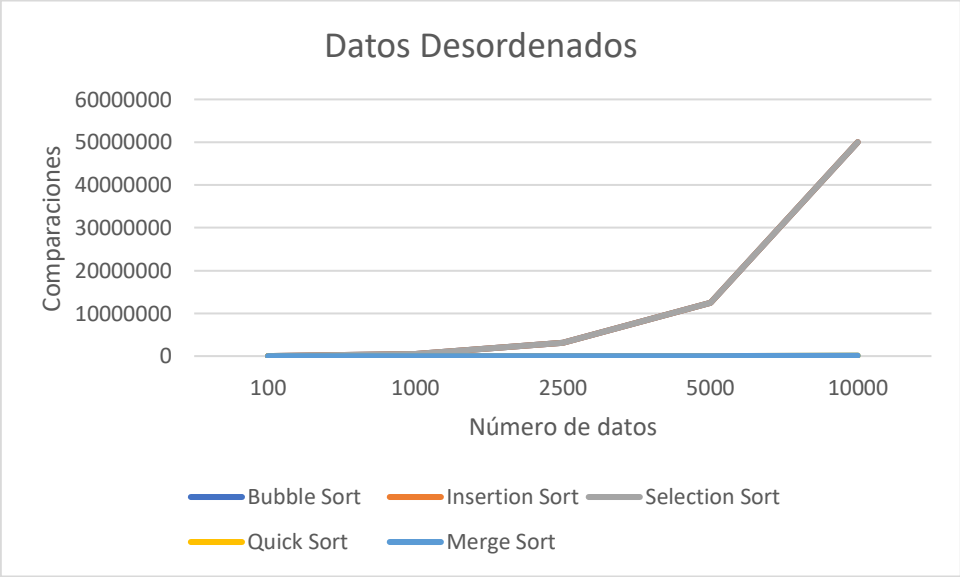
Como se observa en todo momento bubble sort y selection sort, simpre recorren todo el arreglo, realizando las mismas comparaciones; para el caso de los datos ordenados ocurre lo mismo con quick sort; por otro lado merge sort e insertion sort como se vio anteriormente son los más eficaces.

COMPARACIONES	DATOS INVERTIDOS				
	100	1000	2500	5000	10000
BUBBLE SORT	4950	499500	3123750	12497500	49995000
INSERTION SORT	5049	500499	3126249	12502499	50004999
SELECTION SORT	4950	499500	3123750	12497500	49995000
QUICK SORT	4950	499500	3123750	12497500	49995000
MERGE SORT	316	4932	13652	29804	64608



Con datos invertidos, quick sort realiza las mismas comparaciones que bubble y selection sort que si estuvieran ordenados. Insertion sort resulta por poco el más ineficaz.

COMPARACIONES	**DATOS DESORDENADOS**				
	100	1000	2500	5000	10000
BUBBLE SORT	4950	499500	3123750	12497500	49995000
INSERTION SORT	5024	499679	3124284	12500534	50003034
SELECTION SORT	4950	499500	3123750	12497500	49995000
QUICK SORT	646	6700	26832	49736	107328
MERGE SORT	545	8732	25141	55256	120444



Con datos desordenados quick sort y merge sort son las más eficaces. Nuevamente insertion sort es el más ineficaz.