Comparativa de Algoritmos de Ordenación

Algoritmo	**Complejidad Temporal**	**Memoria**	**Estable**	**In-place**	**Situación Recomendada**
Bubble Sort	Peor/Prom: $O(n^2)$, Mejor: $O(n)$	$O(1)$	Sí	Sí	Educativo o listas casi ordenadas
Insertion Sort	Peor/Prom: $O(n^2)$, Mejor: $O(n)$	$O(1)$	Sí	Sí	Listas pequeñas o casi ordenadas
Selection Sort	Siempre $O(n^2)$	$O(1)$	No	Sí	Cuando se desea minimizar escrituras
Merge Sort	Siempre $O(n \log n)$	$O(n)$	Sí	No	Estable y predecible para listas grandes
Quicksort	Prom: $O(n \log n)$, Peor: $O(n^2)$	$O(\log n)$ (pila)	No	Sí	Muy eficiente en la práctica, no estable
Heapsort	Siempre $O(n \log n)$	$O(1)$	No	Sí	Eficiente y con bajo uso de memoria
Shellsort	Aproximadamente $O(n \log^2 n)$	$O(1)$	No	Sí	Buena opción para volúmenes medianos
Counting Sort	$O(n + k)$	$O(n + k)$	Sí	No	Datos pequeños y enteros no negativos
Radix Sort	$O(n \cdot k)$	$O(n + k)$	Sí	No	Claves de longitud fija (números, strings)
Bucket Sort	$O(n + k)$ (ideal)	$O(n + k)$	Sí	No	Datos uniformemente distribuidos