Summary of Sorting Algorithms

		Sorts	Description	Best used for:	Additional memory	Complexity
Simple Non-recursive Small files	Stable	**Bubble**	Compares and swaps adjacent elements. Several passes are needed	General purpose	no	**$O(N^2)$**
		Selection	Finds subsequent maximums	Large records (very little swapping)		
		Insertion	Assumes part of the array is sorted, inserts the next element there.	Almost sorted files		
Advanced Not stable Large files	Recursive	**QuickSort**	Splits the array in two sets and a middle element: smaller to the left, bigger to the right. Then sorts recursively each set	General, commercial, usually very fast **Warning!!!** Run time may increase to $O(N^2)$	no	**O(NlogN)**
	Non-recursive	**HeapSort**	Uses priority heap - the smallest/largest is at the top always	Guaranteed runtime		
	Recursive	**MergeSort**	Splits the array into two, sorts recursively and then merges	Never used for main memory sort. The idea is applied for external sorting	yes	