**10671660**

**INSERTION SORT**

**Insertion sort** is a simple sorting algorithm that builds the final sorted array (or list) one item at a time. It is much less efficient on large lists than more advanced algorithms such as [quicksort](https://en.m.wikipedia.org/wiki/Quicksort), [heapsort](https://en.m.wikipedia.org/wiki/Heapsort), or [merge sort](https://en.m.wikipedia.org/wiki/Merge_sort). However, insertion sort provides several advantages:

Simple implementation: [Jon Bentley](https://en.m.wikipedia.org/wiki/Jon_Bentley_(computer_scientist)) shows a three-line [C](https://en.m.wikipedia.org/wiki/C_(programming_language)) version, and a five-line [optimized](https://en.m.wikipedia.org/wiki/Program_optimization) version[[1]](https://en.m.wikipedia.org/wiki/Insertion_sort#cite_note-pearls-1)

Efficient for (quite) small data sets, much like other quadratic sorting algorithms

More efficient in practice than most other simple quadratic (i.e., [O](https://en.m.wikipedia.org/wiki/Big_O_notation)(*n*2)) algorithms such as [selection sort](https://en.m.wikipedia.org/wiki/Selection_sort) or [bubble sort](https://en.m.wikipedia.org/wiki/Bubble_sort)

[Adaptive](https://en.m.wikipedia.org/wiki/Adaptive_sort), i.e., efficient for data sets that are already substantially sorted: the [time complexity](https://en.m.wikipedia.org/wiki/Time_complexity) is [O](https://en.m.wikipedia.org/wiki/Big_O_notation)(*kn*) when each element in the input is no more than *k* places away from its sorted position

[Stable](https://en.m.wikipedia.org/wiki/Stable_sort); i.e., does not change the relative order of elements with equal keys

[In-place](https://en.m.wikipedia.org/wiki/In-place_algorithm); i.e., only requires a constant amount O(1) of additional memory space

[Online](https://en.m.wikipedia.org/wiki/Online_algorithm); i.e., can sort a list as it receives it

When people manually sort cards in a bridge hand, most use a method that is similar to insertion.

**Worst case performance:** O(*n*2) comparisons and swaps

**Best case performance**: O(n) comparisons, O(1) swaps

**Average performance**: O(n2) comparisons and swaps

**Worst case space complexity**: O(n) total, O(1) auxiliary

**ALGORITHM**

Insertion sort iterates, consuming one input element each repetition and growing a sorted output list. At each iteration, insertion sort removes one element from the input data, finds the location it belongs within the sorted list and inserts it here. It repeats until no input elements remain.