Bubble Sort

you need 3 iterators, I, J and the wall. the process is you bubble the biggest value to the top

In-Place algorithm

uses a small amount of extra memory (doesn't depend on n)

Stable, if it doesn’t change relative order or items

O(n^2) time complexity quadratic

Degrades quickly

Stable vs Unstable Sorting

If the relative order (of duplicated items) is preserved = stabled, otherwise unstable.

Selection Sort

You select either the largest or smallest element in the array, like bubble sort there is a Wall as a limit.

* In-place algorithm (uses small amount of extra mem [doesn’t depend on n])
* Unstable
* Quadratic Time Complexity
* Degrades quickly

Insertion Sort

Uses a wall again set to 1, all elements to the left side of the wall are considered sorted. You save the unsorted value and inserted in and shift and insert it into the sorting array. If next value is greater than the end it will skip the loop and just increment the data.

* In-place algorithm (uses small amount of extra mem [doesn’t depend on n])
* Stable
* Quadratic Time Complexity
* Degrades quickly

Recursion

* Recursive function is a function that calls itself
* Base case to terminate recursive calls
* Classic example is factorial
  + 3! = 3\*2\*1=6
* Depth of recursion is bound; it stays on the stack.
* Can have a stack overflow (CLR will terminate rampant calls)

Shell Sort

* Based on Insertion Sort
* Insertion is fast on pre-sorted arrays
* Basic Idea: pre-sort the input and switch to Insertion Sort
* Gap is used for pre-sorting => Swap distant elements
* Sort starts with a large gap and reduces it gradually
* When gap =1 sort finishes and sort start
* Performance depends on gap value. 99% of cased you can use universal gap. To calculate the max gap < N/3 (where N is the length of the array)
* In-place Algo
* Unstable

Merge Sort (one of the fastest algo)

* Divide and Conquer
* Two phases: splitting and merging
* Splitting is logical: provides an organized way to sequence the merge.
* Split left side first then the right side, then you merge 2 sorted arrays.
* Not an in-place, uses extra mem (n length)
* Stable in a classic implementation
* Linearithmic O(nlogn)

Quick Sort

* Based on Divide and conquer
* Recursive
* Left and Right side contain unsorted elements that are Greater than or less than the pivot
  + Elements < pivot left
  + Elements > go right
* In-Place
* Linearithmic O(nlogn)
* Unstable