

Stable Sorting: two objects with equal keys appear in the same order in the sorted output as they appear in the input data set. Eg, Bubble, Insertion, Merge

10 30 20 60 20
↓

10 20 20 30 60

Unstable Sort: (Visa-versa). Eg Selection, Quick, Heap

InPlace / Non-InPlace: Does not need extra memory (InPlace). Small constant space for variables is allowed.

Sort(---) ← Uses InPlace Sort +
(Quick + Heap Sort + Insertion Sort)
↑
Reduced array size
Larger array size

GFG: Sorting

Stable Sorts: Bubble Sort, Insertion Sort, Merge Sort ---

Unstable Sort: Selection Sort, Quick Sort, Heap Sort ---

→ Use Checklog for that. The complexity is $O(n)$

* Bubble Sort → Stable, InPlace, $O(n^2)$
→ Worst case when array is sorted
→ Or when we copy data to memory it is imp. in EEPROM. If we do more writes Age of ROM reduces.

* Selection Sort → $O(n^2)$
→ Does less memory writes compared to Quicksort, Merge Sort, Insertion Sort, etc. But

→ Cycle Sort is optimal in terms of memory writes.

→ It is basic idea for Heap Sort.

→ NOT Stable, InPlace.

→ Best and worst both case time complexity of selection sort is $O(n^2)$.

* Insertion Sort: $O(n^2)$ worst case, $O(n)$ best case.
Reverse Sorted Sorted Array

→ In place and Stable

→ Best when size of array is small.

→ Used in practice for small arrays (Also in hybrid algorithms such as TimSort and Insertion Sort).
Python Insertion
quick
+
heap

* When array size is large use Quick or Merge sort, but when that array size reduces while sorting switch to Insertion sort.

Questions

① Minimum No. of moves to seat Everyone [leetcode] <https://leetcode.com/problems/minimum-number-of-moves-to-seat-everyone/description/>

② Sort the matrix diagonally [leetcode]

→ <https://leetcode.com/problems/sort-the-matrix-diagonally/submissions/918017762/>

→ Helpful Article → <https://ganeshprasad227.medium.com/sort-2d-matrix-diagonally-coding-interview-matrix-sorting-2f33225801ab>

Lambda

→ <https://en.cppreference.com/w/cpp/language/lambda>