Comparing Sorting Techniques

incapp

Sorting	Based On
Bubble sort:	Swapping
Selection sort:	Swapping
Insertion sort:	Shifting
Radix sort:	Data Copy (Allocation of extra memory)
Counting sort:	Data Copy (Allocation of extra memory)
Merge sort:	Data Copy (Allocation of extra memory)
Quick sort:	Swapping
Heap sort:	Swapping

Comparing Sorting Techniques

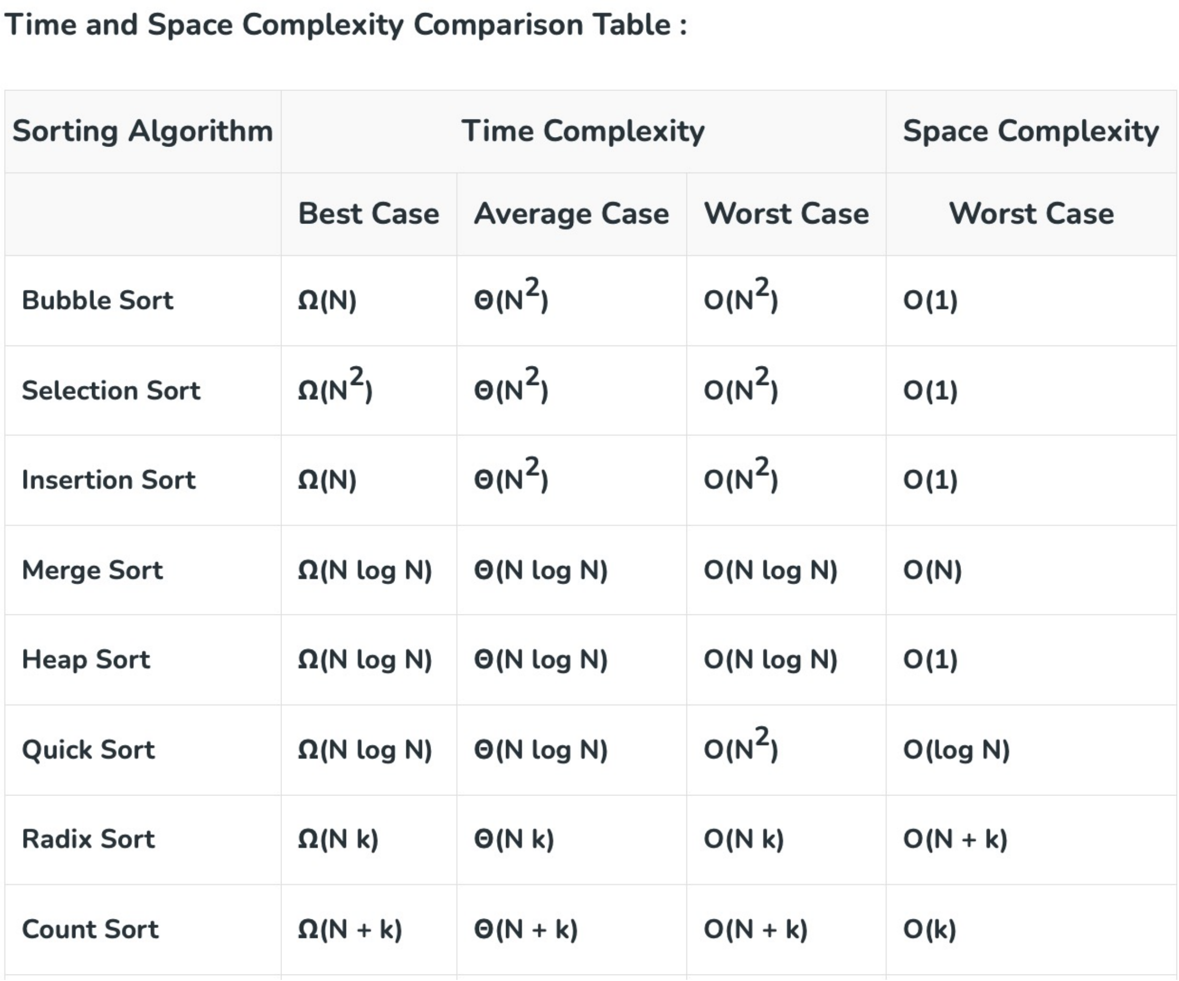
- **Bubble Sort:**
  - Compares elements to bubble up the maximum to the end.
  - Sorting a deck of playing cards.
  - Organizing a small list of names in alphabetical order.
- **Selection Sort:**
  - Compares elements to find the minimum element in the unsorted part and places that element in the
  - sorted part.
  - Finding the top scorers in a sports competition.
  - Sorting a list of students based on their grades.
- **Insertion Sort:**
  - Compares elements to determine the position of an element in the partially sorted array.
  - Inserting new elements into a sorted database.
  - Sorting a hand of playing cards as new cards are dealt.
- **Merge Sort:**
  - Compares elements of two sorted halves to merge them into the final sorted array.
  - Merging and sorting multiple sorted lists or arrays.
  - Sorting large datasets in external storage systems.

Comparing Sorting Techniques

- **Quick Sort:**
  - Compares elements to partition the unsorted array into two different halves around the pivot.
  - Sorting a large list of names or phone numbers in a phonebook.
  - Implementing sorting in databases and search engines.
- **Heap Sort:**
  - Compares elements during the heapify process to place the elements at the correct position in the sorted array.
  - Priority queue implementation in scheduling algorithms.
  - Sorting processes based on their priority in an operating system.
- **Radix Sort:**
  - Sorting student records based on multiple attributes such as name and age.
  - Sorting and organizing files based on their file extension.
- **Counting Sort:**
  - Sorting and counting votes in an election.
  - Analyzing the frequency of words in a text document.

Quicksort is one of the most efficient sorting algorithms

Comparing Sorting Techniques



Time and Space Complexity Comparison Table :

Sorting Algorithm	Time Complexity			Space Complexity
	Best Case	Average Case	Worst Case	Worst Case
Bubble Sort	Ω(N)	Θ(N^2)	O(N^2)	O(1)
Selection Sort	Ω(N^2)	Θ(N^2)	O(N^2)	O(1)
Insertion Sort	Ω(N)	Θ(N^2)	O(N^2)	O(1)
Merge Sort	Ω(N log N)	Θ(N log N)	O(N log N)	O(N)
Heap Sort	Ω(N log N)	Θ(N log N)	O(N log N)	O(1)
Quick Sort	Ω(N log N)	Θ(N log N)	O(N^2)	O(log N)
Radix Sort	Ω(N k)	Θ(N k)	O(N k)	O(N + k)
Count Sort	Ω(N + k)	Θ(N + k)	O(N + k)	O(k)