Difference between

Comparison (QuickSort) **based**

and

Non-Comparison (Counting Sort) **based**

Sorting Algorithms?

Comparison Sorting

- A **comparison sort** is a type of sorting algorithm that only reads the list elements through a single comparison operation (often a "less than or equal to" operator or a three-way comparison) and determines which of two elements should occur first in the final sorted list.
- What is common to all these algorithms?
  - Make **comparisons** between input elements like:

$$a_i < a_j, \quad a_i \leq a_j, \quad a_i = a_j, \quad a_i \geq a_j, \quad \text{or} \quad a_i > a_j$$

Sorting Algorithms	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)
Quick Sort	Ω(N log N)	Θ(N log N)	O(N^2)	O(N)
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)

Non Comparison Sorting

- There are some sorting algorithms that perform sorting without comparing the elements rather by making certain assumptions about the data. They are called the non comparison sorting.
- Non comparison sorting include:

1. Counting sort (indexes using key values)
2. Radix sort (examines individual bits of keys)
3. Bucket sort (examines bits of keys)

- These are Linear sorting algorithms. Linear sorts are NOT "comparison sorts".
- They make certain assumptions about the data. These type of sorting algorithm does not need to go through the comparison decision tree.

- Many times we have restrictions on our keys
  - Deck of cards: Ace->King and four suites
  - Social Security Numbers
  - Employee ID's

- We will examine three algorithms which under certain conditions can run in **O(*n*)** time.
  - Counting sort
  - Radix sort
  - Bucket sort

Sorting Algorithms	Special Input Condition	Time Complexity			Space Complexity
		Best Case	Average Case	Worst Case	Worst Case
Counting Sort	Each input element is an integer in the range 0- K	Ω(N + K)	Θ(N + K)	O(N + K)	O(K)
Radix Sort	Given n digit number in which each digit can take on up to K possible values	Ω(NK)	Θ(NK)	O(NK)	O(N + K)
Bucket Sort	Input is generated by the random process that distributes elements uniformly and independently over the interval [0, 1)	Ω(N + K)	Θ(N + K)	O(N^2)	O(N)