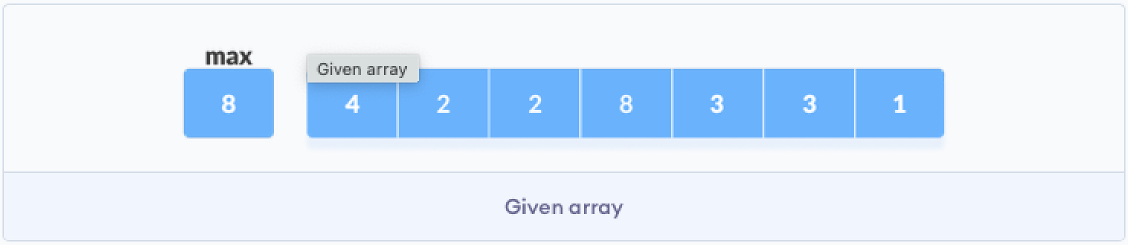
Counting Sort

💡 ***Counting Sort** is a **non-comparison-based** sorting algorithm. The basic idea behind **Counting Sort** is to count the **frequency** of each distinct element in the input array and use that information to place the elements in their correct sorted positions.*

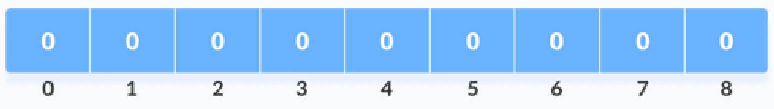
Algorithm Steps

1. Find out the maximum element (let it be `max`) from the given array.



Given array

2. Initialize an array of length `max+1` with all elements 0. This array is used for storing the count of the elements in the array.

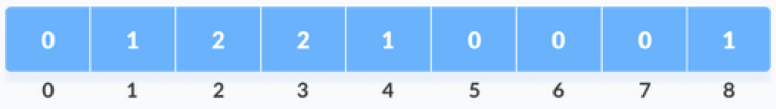


Count array

3. Store the count of each element at their respective index in `count` array.

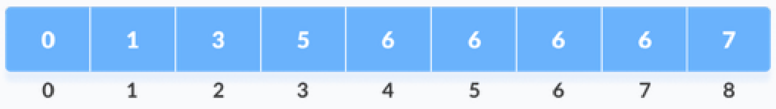
   For example: if the count of element 3 is 2 then, 2 is stored in the 3rd position of `count` array. If element "5" is not present in the array, then 0 is stored in 5th position.

3rd party ad content



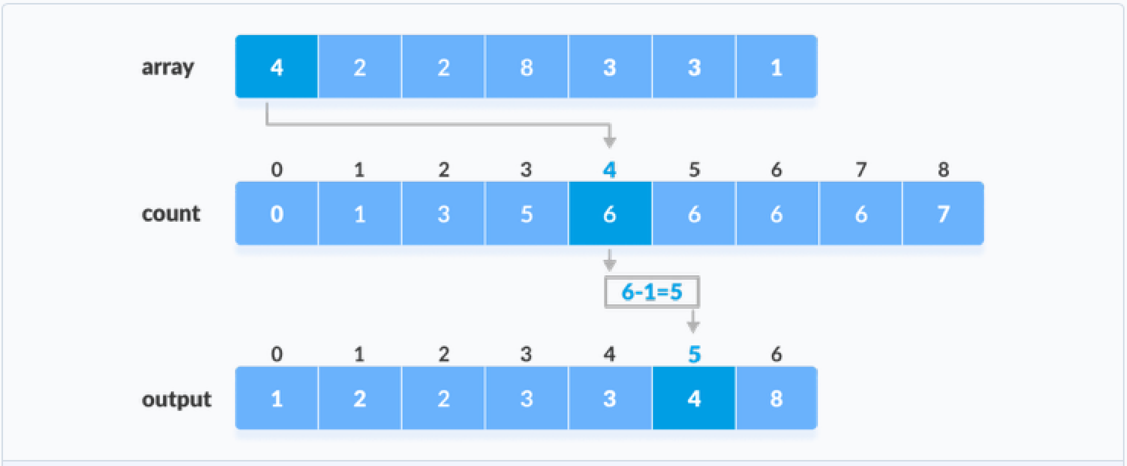
Count of each element stored

4. Store cumulative sum of the elements of the count array. It helps in placing the elements into the correct index of the sorted array.



Cumulative count

5. Find the index of each element of the original array in the count array. This gives the cumulative count. Place the element at the index calculated as shown in figure below.



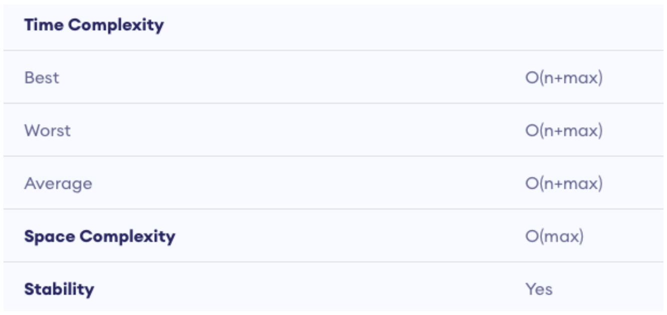
Counting sort

6. After placing each element at its correct position, decrease its count by one.

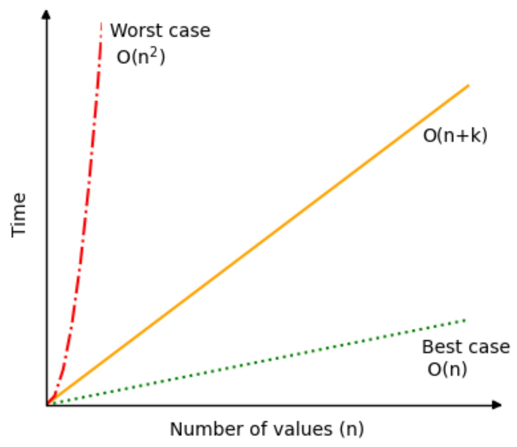
Pseudocode

```
countingSort(array, size)
  max <- find largest element in array
  initialize count array with all zeros
  for j <- 0 to size
    find the total count of each unique element and
    store the count at jth index in count array
  for i <- 1 to max
    find the cumulative sum and store it in count array itself
  for j <- size down to 1
    restore the elements to array
    decrease count of each element restored by 1
```

Analysis



Time Complexity	
Best	O(n+max)
Worst	O(n+max)
Average	O(n+max)
Space Complexity	O(max)
Stability	Yes



Overall complexity

= `O(max)+O(size)+O(max)+O(size)` = `O(max+size)`

- **Worst Case Complexity:** `O(n+max)`
- **Best Case Complexity:** `O(n+max)`
- **Average Case Complexity:** `O(n+max)`

In all the above cases, the complexity is the same because no matter how the elements are placed in the array, the algorithm goes through `n+max` times.

Advantage of Counting Sort

- Counting sort generally performs faster than all comparison-based sorting algorithms, such as merge sort and quicksort, if the range of input is of the order of the number of input.
- Counting sort is easy to code
- Counting sort is a **stable algorithm**.

Disadvantage of Counting Sort

- Counting sort doesn't work on decimal values.
- Counting sort is inefficient if the range of values to be sorted is very large.
- Counting sort is not an **In-place sorting** algorithm, It uses extra space for sorting the array elements.

Applications of Counting Sort

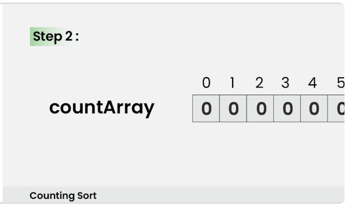
- It is a commonly used algorithm for the cases where we have limited range items. For example, sort students by grades, sort a events by time, days, months, years, etc
- It is used as a subroutine in Radix Sort
- The idea of counting sort is used in Bucket Sort to divide elements into different buckets.

📌 Java implementation can be found under Implementation_Java folder

📚 References

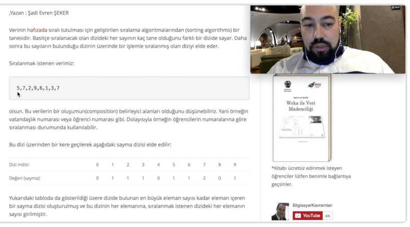
Counting Sort (With Code in Python/C++/Java/C)
Counting sort is a sorting algorithm that sorts the elements of an array by counting the number of occurrences of each unique element in the array and sorting them according to the keys that are small integers. In this tutorial, you will understand the working of counting sort with working code in C, C++, Java, and Python.
https://www.programiz.com/dsa/counting-sort

Counting Sort - Data Structures and Algorithms Tutorials - GeeksforGeeks
Counting sort is a sorting technique based on keys between a specific range. It works by counting the number of objects having distinct key values
https://www.geeksforgeeks.org/counting-sort/



W3Schools.com
W3Schools offers free online tutorials, references and exercises in all the major languages of the web. Covering popular subjects like HTML, CSS, JavaScript, Python, SQL, Java, and many, many more.
https://www.w3schools.com/dsa/dsa_algo_countingsort.php

Counting Sort (Sayarak Sıralama) Algoritma Analizi 6
Algoritma analizi açısından sayarak sıralama olarak geçen (counting sort) algoritmanın çalışmasını anlatıp analizini yapıyoruz, en iyi, en kötü ve ortalama durum analizlerine bakıyoruz.
https://www.youtube.com/watch?v=fkX5Zy1qL7Y



✍🏼 Author → Serhat Kumas

https://www.linkedin.com/in/serhatkumas/

SerhatKumas - Overview

Computer engineering student who loves coding in different fields instead of focusing on a one spesific area. - SerhatKumas

https://github.com/SerhatKumas