Count Sort Algorithm:

* Non-comparison algo.
* Time Complexity — Linear
* Single digit whole numbers. (0 – 9)

Steps:

(I) Find the max

(II) Create a Count Array [0 – max] [0 – 6]

(III) Calculate the occurrence of each element of the array.

(IV) Cumulative Count

(V) O/p array

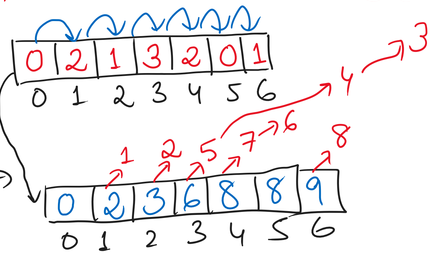
(VI) Start from the end

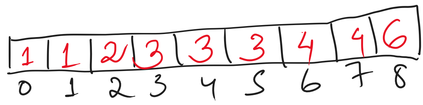
(VII) Copy the output back to the input array

input → 1 3 2 3 4 1 6 4 3

O(N+K)

Count →





Radix Sort Algorithm:

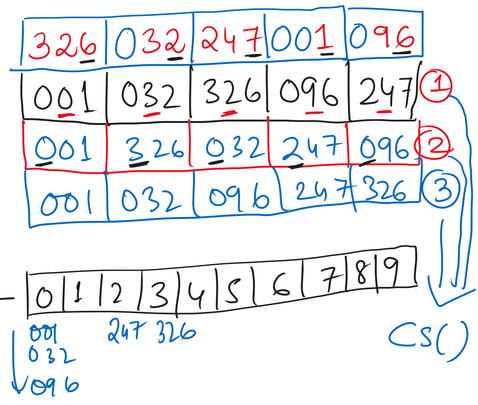
(I) Non-comparison algo

(II) Multi-digit numbers

(III) Constant length Strings

* Bucket Sort [0-9]
* Max → 326 → (3) passes

H/W O(n+k)



User-defined data Types:

Structures & Unions

* Can be used to represent a real entity with multiple values.
* Only difference is the access & the memory allocation.

Person → name, age, aadharID

Struct: V1 — M1, V2 — M2, V3 — M2

Size = Σ S1+S2+S3

Union: V1, V2, V3 → M

size = largest

```
Struct Person {
    char[20] name;
    int age;
    int ID;
};
```

Size = 20+4+4 = 28 bytes

```
union Person {
    char[20] name;
    int age;
    int ID;
};
```

20 bytes

Assign and display