Count Sort

Sort array of even & odd

Merge Sort

Stable & In place Sorting

Q. Find smallest no. that can be formed by rearranging digits given in an array.

A = < 6, 3, 4, 2, 7, 2, 1 >

ans → < 1, 2, 2, 3, 4, 6, 7 >

A = < 4, 2, 7, 3, 9, 0 >

ans → < 0, 2, 3, 4, 7, 9 >

Approach 1: Sort the array

Arrays.sort()

↓

TC: O(N log₂N)

Approach 2: Can we use the info that digits are 0 → 9

< -, -, -, -, -, -, -, - >

0 00... 111... 222... 3..... 9

freq of every digit

0 → -

1 → -

2 → -

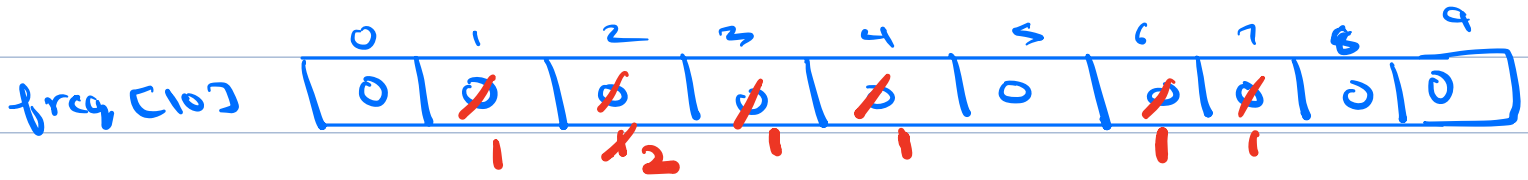
⋮

9 → -

① Count freq of all the digits int freq[10]

② Use freq array to fill the original array

A = < 6, 3, 4, 2, 7, 2, 1 >



A = < 6, 3, 4, 2, 7, 2, 1 >
0 1 2 3 4 5 6
1 2 2 3 4 6 7

Sort based on counting frequency

↓

Count sort

```
// int A[], int N
int freq[10] = <0>
① for (i=0 ; i<N ; i++) {
      freq[A[i]] = freq[A[i]] + 1
  }

int k = 0
② for (dig = 0 ; dig <= 9 ; dig++) {
      for (cnt = 1 ; cnt <= freq[dig] ; cnt++) {
          A[k] = dig      k++
      }
  }
return A
```

N

10

2

dig → cnt
0 → for
1 → for
2 → for

TC : $O(N + 10 + N)$

$\simeq O(N)$

SC : $O(10) \simeq O(1)$

↓

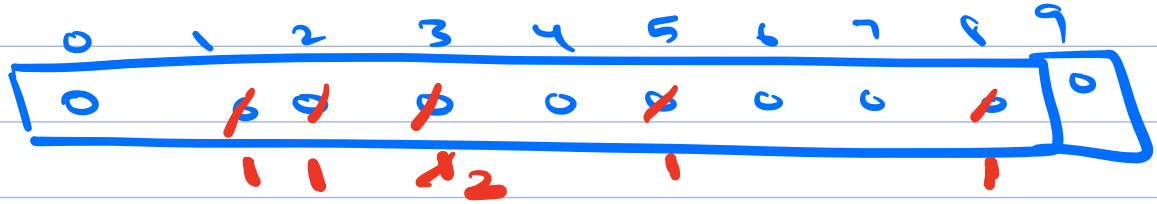
No. of digits (0-9)

3 → ⋮ 9

N

$A = [1, 3, 8, 2, 3, 5]$

freq[10] =



$A = [1, 3, 8, 2, 3, 5]$

1 2 3 3 5 8

Will count sort work if range of A[i] is more than 10^9 ?

$< 10^9, 10^7 + 2, 10^8, 10^9 >$

$10^9 \rightarrow 2$

$10^7 + 2 \rightarrow 1$

$10^8 \rightarrow 1$

data → 0 to 10^9

int freq[10^9+1]

int → 4 B

10^9 integers → 4 B × 10^9 = 4 GB

Count sort works until till 10^6

Range → 0 to 10^6

int freq[10^6]

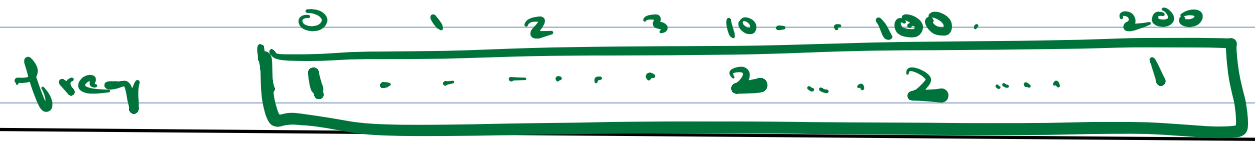
↓

4×10^6 B = 4 MB

< 10, 0, 100, 200, 100, 10 >

range → 0 to 200

int freq[201]



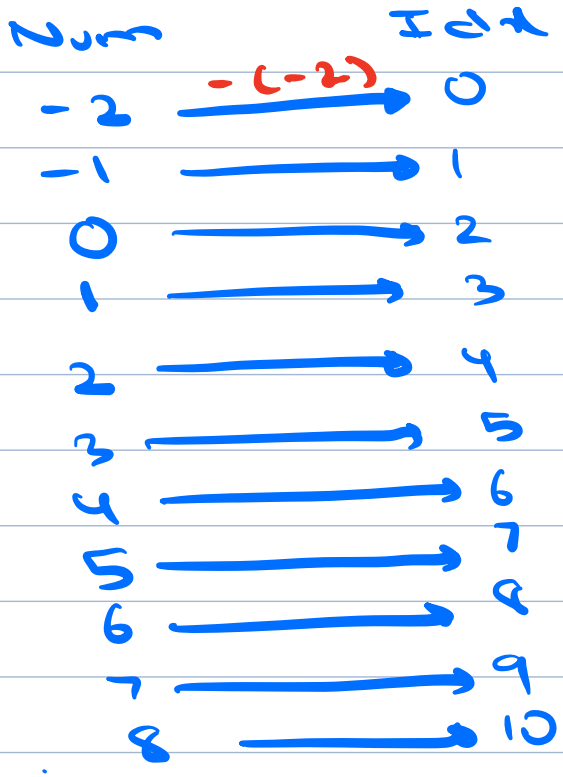
Case II A = [-2 3 8 3 -2 3]

Range → -2 to 8

int freq[11]

No. of el in range =

8 - (-2) + 1 = 11



Num - (-2) = Idx

Num - Min = Idx

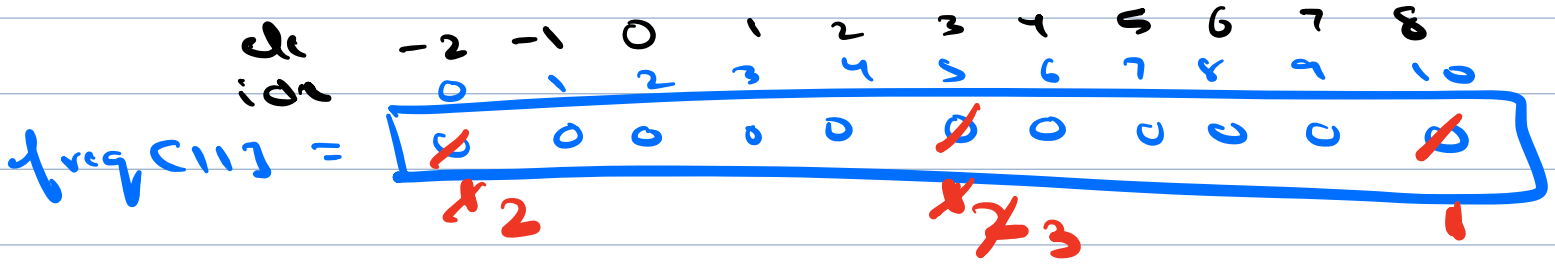
Idx = Num + Min

A = [-2 3 8 3 -2 3]

Range → -2 to 8

int freq[11]

Min = -2 Max = 8



A = [-2 -2 3 3 3 8]

① Iterate in array and get min and max → N

② int freq[max - min + 1] = <0>

for (int i = 0 ; i < N ; i++) → N

idx = A[i] - min

freq[idx] ++

int k = 0

for (int i = 0 ; i < max - min + 1 ; i++) → Range

num = i + min → N

```
for (cnt = 1 ; cnt <= freq[i]; cnt++){
    A[k] = num   k++
}
return A
```

TC: O(3N + Range)

≃ O(N + Range)

SC: O(Range)

freq[max-min+1]

10:06