Sorting

13 August 2023 13:07

- ✓ Bubble sort
- Selection sort
- Insertion sort

Time → $O(N^2)$

Iterative

- ✓ Merge sort
- ✓ Quick sort (in place)

Time → $O(N \log N)$

recursive algo.

Bubble sort

13 August 2023 13:10

Take the largest element towards the end.

arr = [5, 2, 4, 1, 3]

→ [2, 5, 4, 1, 3]

→ [2, 4, 5, 1, 3]

→ [2, 4, 1, 5, 3]

→ [2, 4, 1, 3, 5]

1st iteration

unsorted | sorted

→ [2, 1, 4, 3, 5]

→ [2, 1, 3, 4, 5]

2nd iteration

unsorted | sorted

(5) → 4 times

→ [1, 2, 3, 4, 5]

3rd iteration

unsorted | sorted

→ [1, 2, 3, 4, 5]

4th iteration

sorted

```
//iterations
for(i=0; i<=n-2)
//pairwise comparisons
  for(j=0; j<n-1)
```

swapping logic

Time → $O(N^2)$

Space → $O(1)$

Java

C++

```
sort(arr, arr+n);
```

Java

```
Arrays.sort(arr);
```

Merge Intervals

13 August 2023 13:28

intervals[] = [5,8], [1,3] [9,10], [2,4], [10,12]

ans = [1,4], [5,8], [9,12]

start, end

Brute force → $O(N^2)$ → Time

$O(1)$ → Space

→ [1,3], [2,4]

→ [1,4]

intervals[] = [5,8], [1,3], [10,15), [2,4], [12,13]

Sorting → $O(N \log N)$ based on start time.

sort[] = [1,3], [2,4], [5,8], [10,15], [12,13], [18,20]

start 1 = 1, 5, 10, 18

end 1 = 3, 4, 8, 15, 20

start 2 = 5, 10, 12, 18

end 2 = 4, 8, 15, 13, 20

if(s2 <= e1)

// overlapping

[1,4], [5,8), [10,15], [18,20]

for(i=1; i<n)

$O(N \log N) + O(N)$

$O(N \log N)$

= O(1) ignore

O(N) if not.

Strings

13 August 2023 13:54

Palindrome

madam

ffg

racecar

radar

racecar

true

madem

false

Ascii table

A - 65

B - 66

C - 67

a - 97

b - 98

c - 99

Valid anagrams

13 August 2023 13:59

cat - tac

silent - listen

race - care

state - taste

s1 = a b a c d a → 3 times

s2 = a a c d b → 2 times

false

① Two loops. $O(N^2)$

② Sort. → $O(N \log N)$

③ Array / Map.

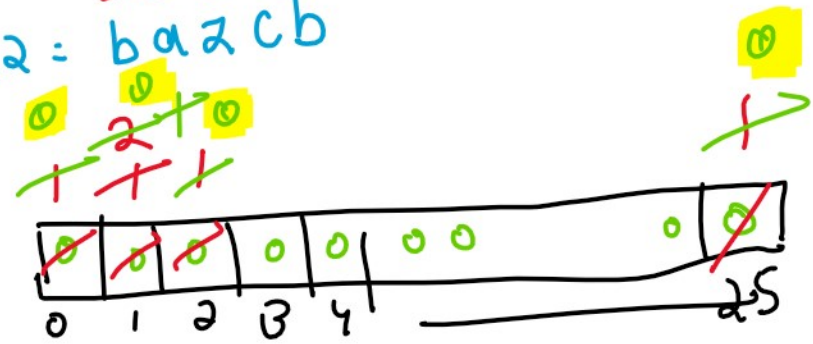
(length should be same)

Map → Array.

0 → a
1 → b
2 → c
3 → d
|
25 → z

s1 = a b c b z

s2 = b a z c b



1. traverse first ~~array~~ string and increment the freq.
2. traverse second array and decrement the freq.
3. check all the array freq. should be zero.

s1 = a b c c d c (N)

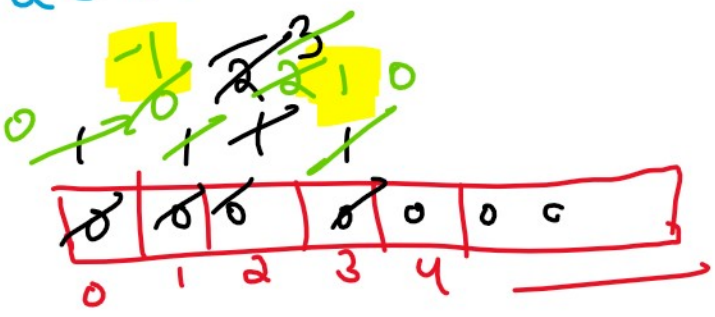
s2 = a c c b b d (N)

Time → O(N) + O(N) = O(N)

Space → O(1)

O(N) | O(1)

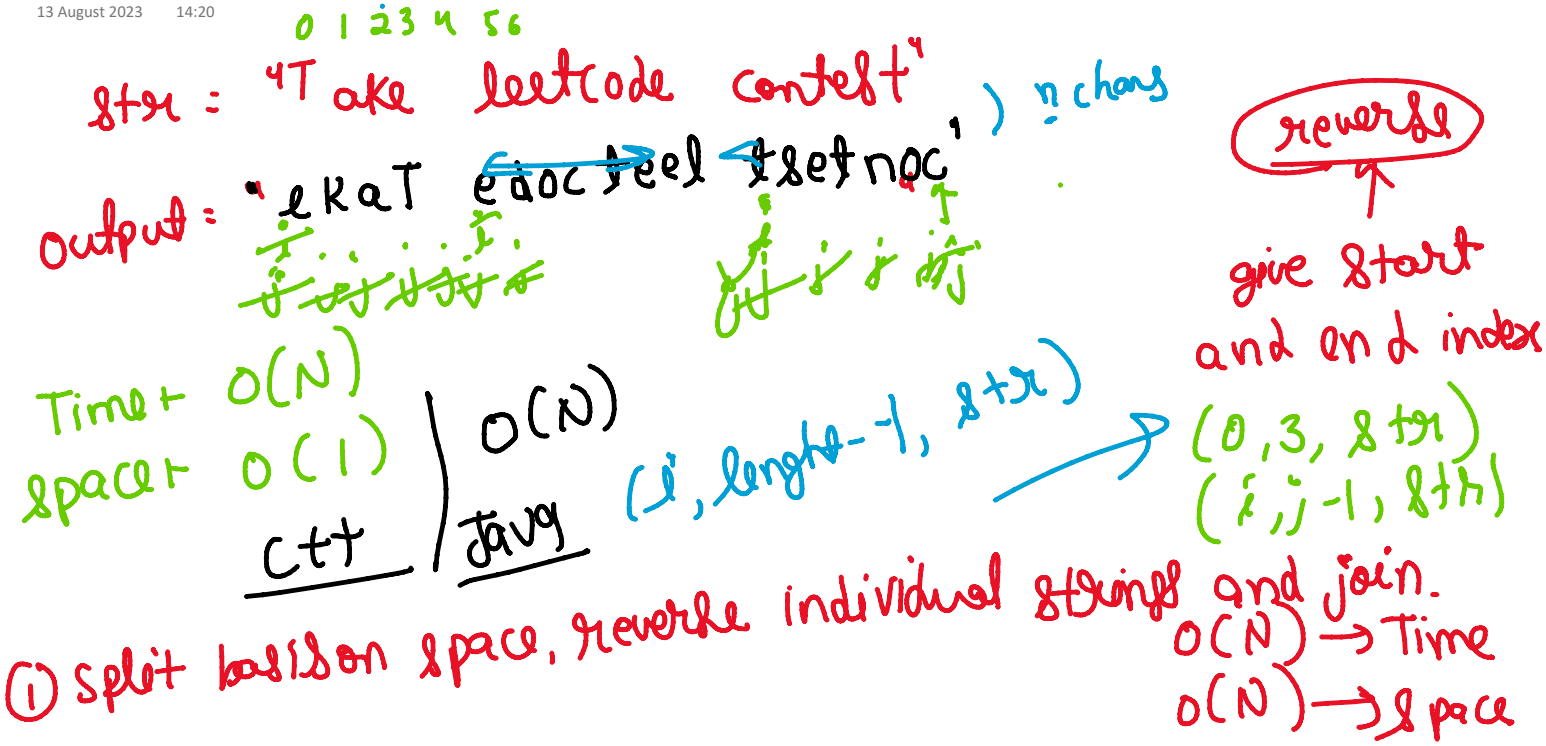
s2 = a c c b b d



O(N) | O(1)

Reverse words in a string

13 August 2023 14:20



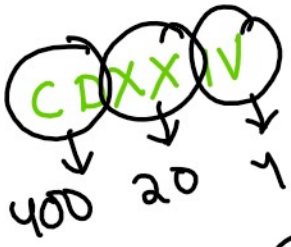
Roman to Integer

13 August 2023 15:03

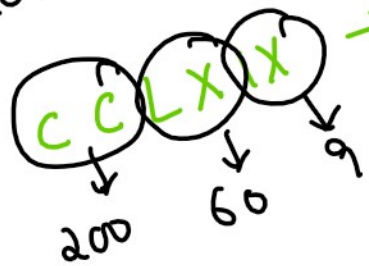
I - 1
II - 2
III - 3
IV - 4
V - 5
VI - 6
VII - 8
IX - 9
X - 10

XII - 12
XVI - 16
XIX - 19
XLIV - 44
XL - 40

1 - I
5 - V
10 - X
50 - L
100 - C
500 - D
1000 - M



→ 424



→ 269

→ LXVIII → 68

50+10+5+1+1+1 = 68

→ CDXXIV →

(500-100)+10+10+(5-1)

= 400+20+4 = 424

→ MCMXCVII → 1997

array
map
if-else
switch

if the next value is greater, then decrement otherwise increment.

1000+100-10+10-1+1 = 1997

→ M CM X C VII → ?

req = 1000 + (1000 - 100) + (100 - 10) + 5 + 1 + 1 = 1997

Time: O(N)
Space: O(1)

Longest palindromic substring

13 August 2023 15:20

ans = pp

str: apple

pal sub - pp, a, p, l, e

str = a b l e v e l a b

ans: level,

str = forgeeksskeegfor

ans = geekssskeeg, ee, ss

Brute force: generate all possible substring & check for palindrome

Time: $O(N^3)$

Space: $O(1)$

0	1	2	3	4	5	6	7	8
a	b	l	e	v	e	l	a	b

(odd length)

str =

ans: ~~a~~ level

start char = 2

max len = ~~1~~ 5

compute: $r - l - 1$

$= 7 - 1 - 1 = 5$

str.substr(,)

l--

Str.substr(,)

left and right to expand in both dirn

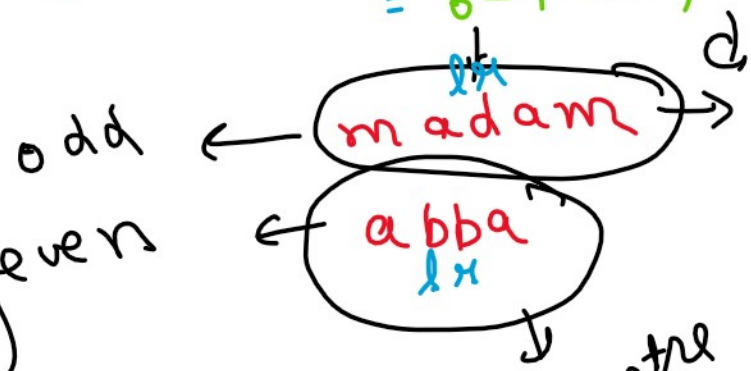
l - -
r ++

Time $O(N^2)$
Space $O(1)$
(even length)

ans = zyyz

palindrome → odd length
palindrome → even length

len = r - l - 1
= 6 - 1 - 1 = 4



for(i=0; i<n; i++)

$O(N^2)$
$O(1)$

maxLen = ~~1~~ ~~4~~ 5

startIdx = ~~0~~ ~~0~~ 4
(l+1)

r-l+1
r-l-1

ans = ~~c~~ abba

.substr(startIdx, maxLen)

maxLen = 4 - (-1) - 1 = 4+1-1-4
= 5-1=5

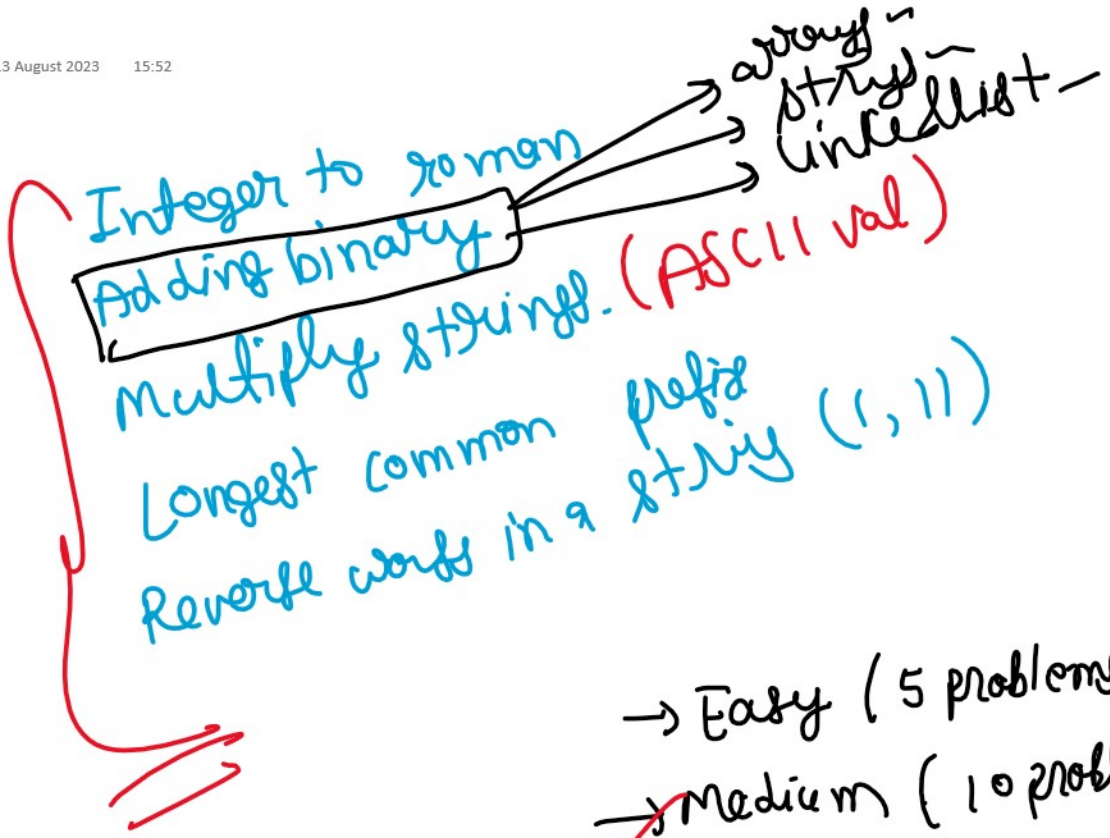
substr (start
(4, 5)

cwordLen = 4 - (-1) = 5
= 9 - 3 - 1 = 5

r - l - 1

l - -
r ++

13 August 2023 15:52



90 %.

50 %.

→ Easy (5 problems) 10 %.

→ Medium (10 problems)

→ Hard (3-4) prob

Me