Minimum Size Subarray Sum

Saturday, 18 September 2021 10:31 AM

target = 7, nums = [2,3,1,2,4,3]

0	1	2	3	4	5
2	3	1	2	4	3

left

right

left pointer

right pointer =

result: ~~∞~~ ~~4~~ ~~3~~ 2

sum: ~~0~~ ~~2~~ ~~5~~ ~~6~~ ~~6~~ ~~10~~ ~~7~~ ~~6~~ ~~2~~ 7 ~~3~~

① Check if element is greater than target? → if yes return 1.

② If not then add it in current sum.

③ Check if left most element is skipable, so that subarray length can reduce.

If skipable.

→ Set result with sum

→ reduce current sum

→ move left pointer

Maximum Average Subarray I

Saturday, 18 September 2021 10:31 AM

① Sum of initial (k+1) size window

while (i < n)

sum += arr[i];

cur.Avg = sum/(i - j);

res = math.max(res, cur.Avg);

sum -= arr[i];

i++;

j++;

return res;

Calculate avg

add next

remove prev

make a check for left window

① add new Elem

② calculate avg

③ remove left elem

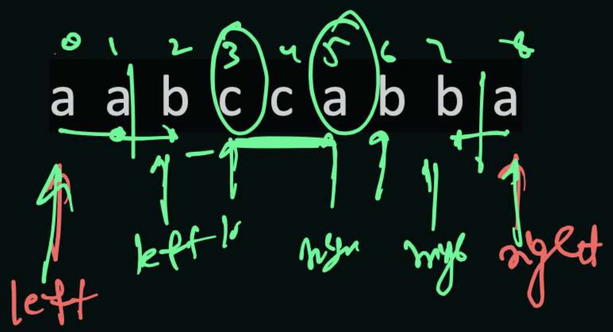
i++;

j++;

no extra check is required

Minimum Length of String After Deleting Similar Ends

Saturday, 18 September 2021 10:31 AM



left=0
right=n-1

while (left < right && str[left] == str[right]) {

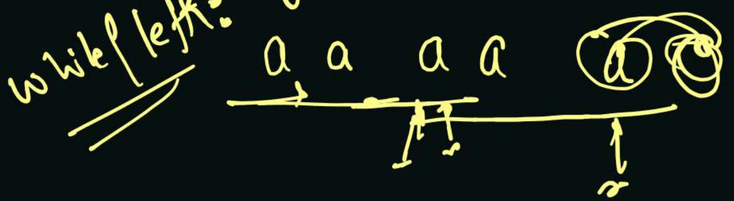
ch = str[left];

// move left ahead

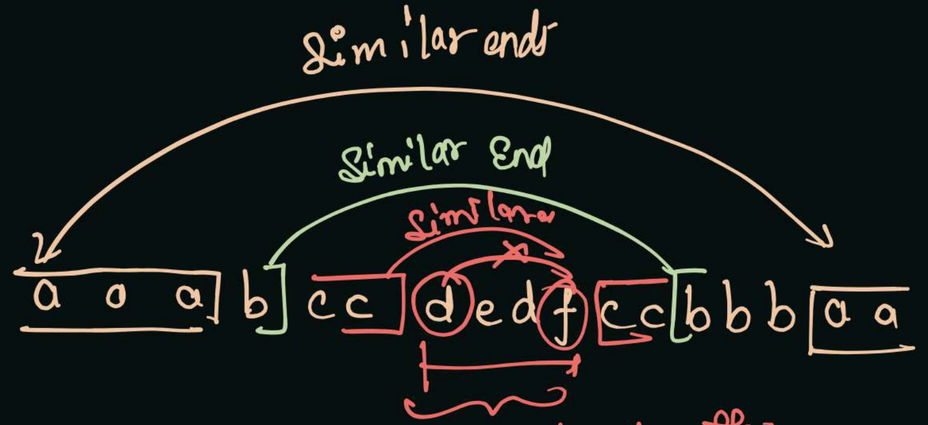
// move right in backward

} (left < right && str.charAt(left) == ch)

while (left == right)



ch = 'a'



ch = a b

return right - left + 1

minLength = 0

right - left + 1 == 0

Find All Duplicates in an Array

Saturday, 18 September 2021 10:31 AM

elements → [1 to n]

val = true

Single or Twice

$O(n^2)$ → Brute force

$O(n \log n)$ → sorting Brute force

0	1	2	3	4	5	6	7
[4,	3,	2,	7,	8,	2,	3,	1]
-4	-3	-2	-7			-3	-1

Optimised soln

indx: 3 2 1 6 7 1 2 0

[2, 3] → result

```
indx = Math.abs(nums[i]) - 1;
val = nums[indx]
if(val < 0){
    // add (indx+1) in result
    res.add(indx+1);
}else{ // malc :(
}
}
```