SLIDING WINDOW TECHNIQUE 🪟🚀

Master DSA Patterns | Become a Pro Coder

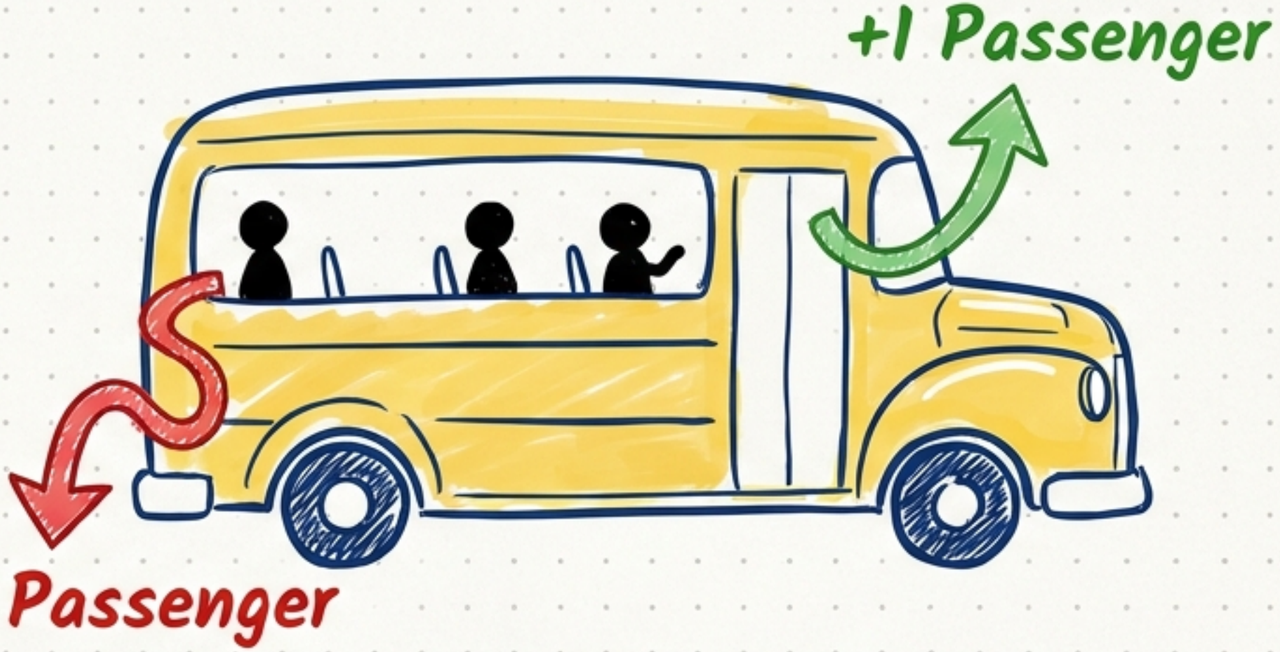
Notes made easy!

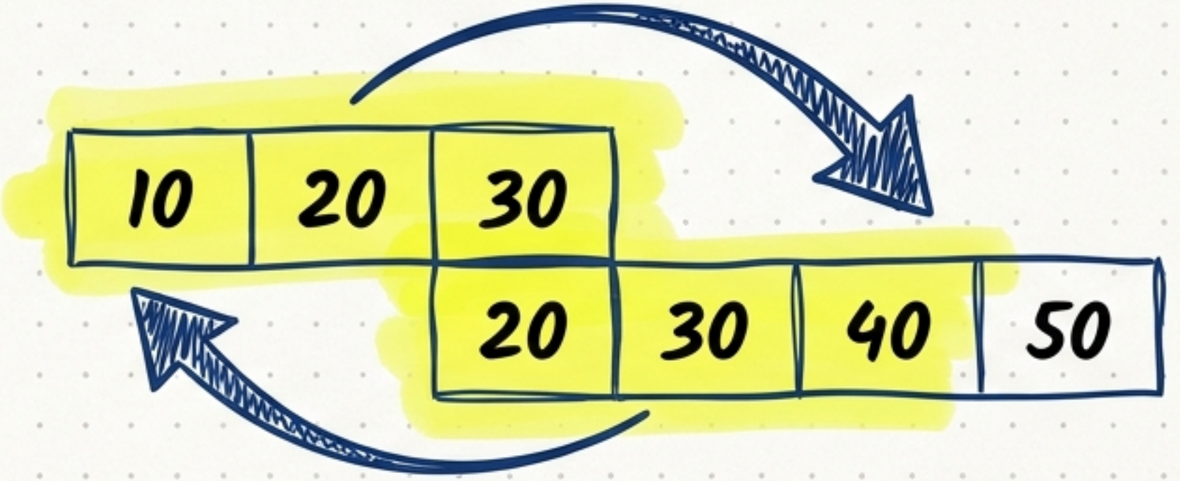
For beginners & interview prep. 100

What is Sliding Window? 🤔

Jab humein ek Array ya String mein continuous elements (Subarray) par kaam karna hota hai, tab hum ek Window banate hain.

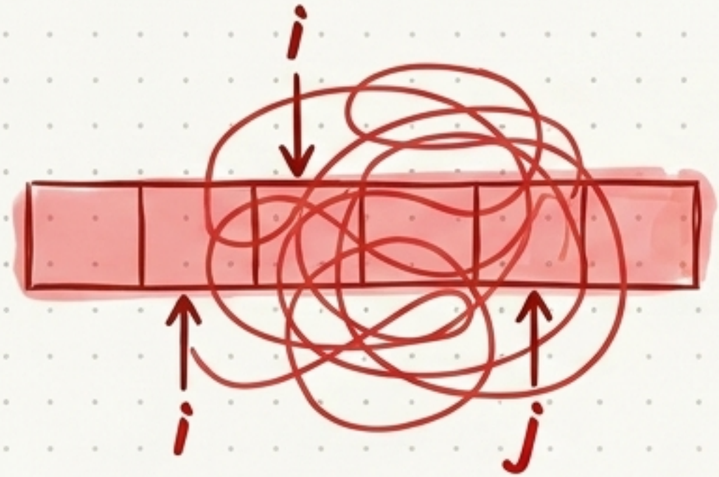
- Pura array wapas traverse karne ke bajaye, hum **window ko aage slide** karte hain.
- Ek element cut hota hai, ek naya add hota hai!





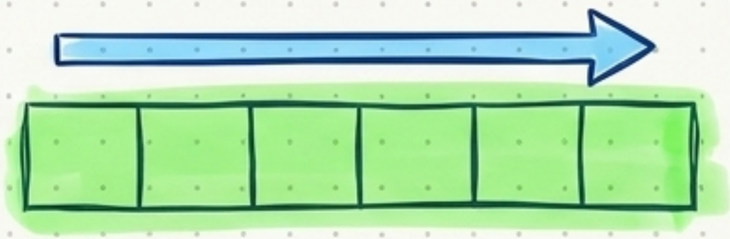
Nested Loops (Brute Force) ko Single Loop mein convert karta hai!

Brute Force vs Optimized Pattern ⚔️



TLE

- Har subarray ko baar-baar calculate karna.
- Loops inside Loops: for(i) → for(j)
- Time Complexity: $O(N^2)$ or $O(N \times K)$ ❌ (Interview mein fail)



- Sirf previous result mein 1 element minus, 1 element plus karna!
- Single traversal: Sirf ek while loop.
- Time Complexity: $O(N)$ ✅ (Interviewer khush!)

Magic of Sliding Window = Repetitive work ko eliminate karna!

Types of Sliding Window Problems

Fixed Size Window

- **Identify:** Window ka **size (K)** pehle se diya hota hai.
- **Example:** Find max sum of subarray of size 3.

Variable Size Window

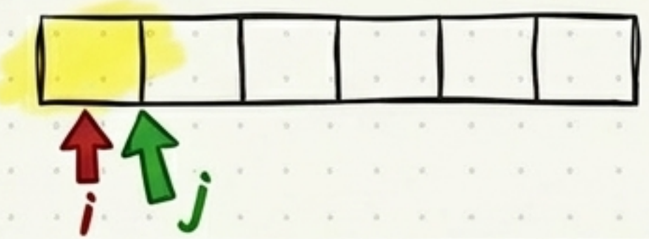
- **Identify:** Window ka **size** fix nahi hota. **Condition** di hoti hai.
- **Example:** Find minimum subarray size whose sum is ≥ 7.

Pro-Tip: Question read karke sabse pehle identify karo: Size Fixed hai ya Condition di hai?

Fixed Window ka Brahmastra (Steps) 🛠️

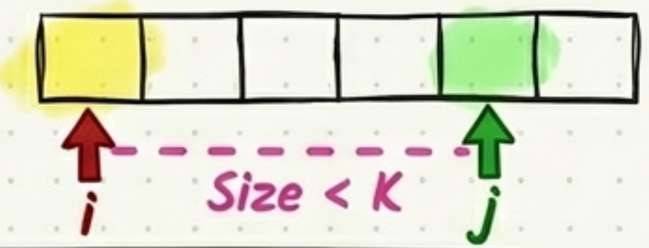
Step 1: Start the Window.

Pointers $i = 0, j = 0$ set karo.
Sum/Count variables initialize karo.



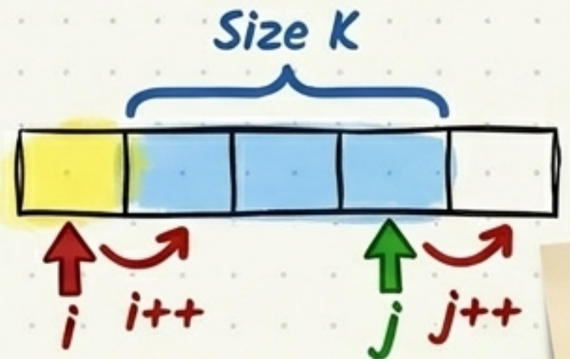
Step 2: Expand to Size K.

j ko aage badhao aur calculations add karte jao
jab tak window size K na ban jaye $(j - i + 1 == K)$.



Step 3: Maintain Size & Slide!

Jaise hi size K ho jaye:
a) Apna answer update karo (e.g., max/min store karo).
b) i wale element ka contribution remove karo.
c) Window ko aage khiskao (i++, j++).



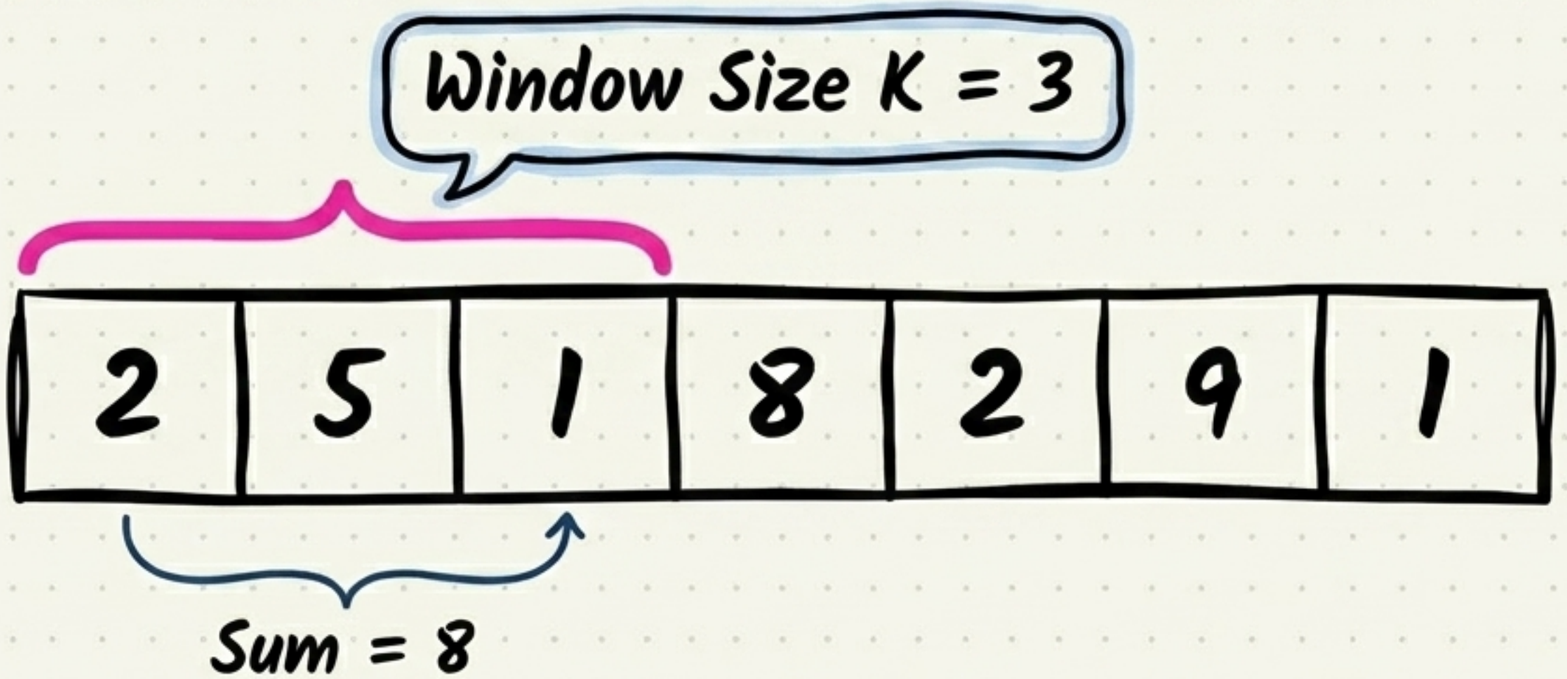
Problem 1 - Max Sum of Subarray of Size K 🎯

Ek array diya hai, aur size K diya hai.
Us subarray ka maximum sum nikalna hai.

Goal: Find the contiguous subarray of size K with the maximum sum.

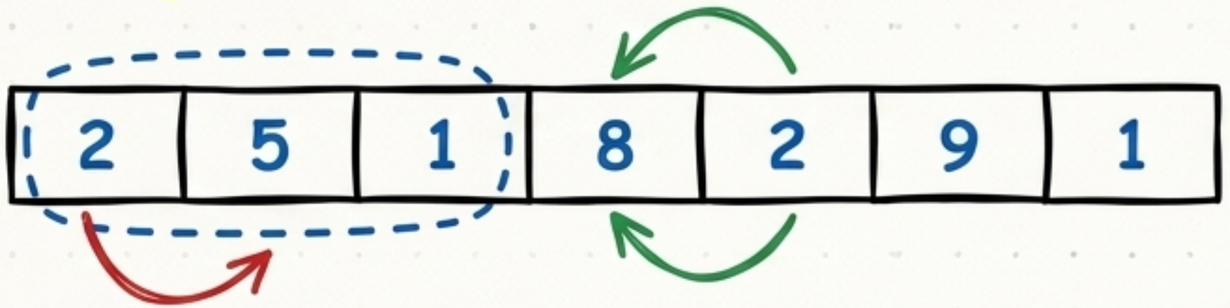
✅ Subarray find karna hai?
✅ Window ka size K fix hai?

→ **Conclusion:** Fixed Window pattern apply hoga! ✅

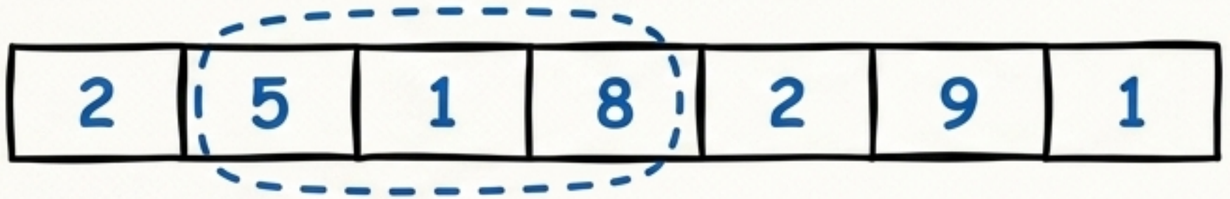


Dry Run Ka Jadoo ✨

The Golden Formula: New Sum = Old Sum - arr[i] + arr[j]

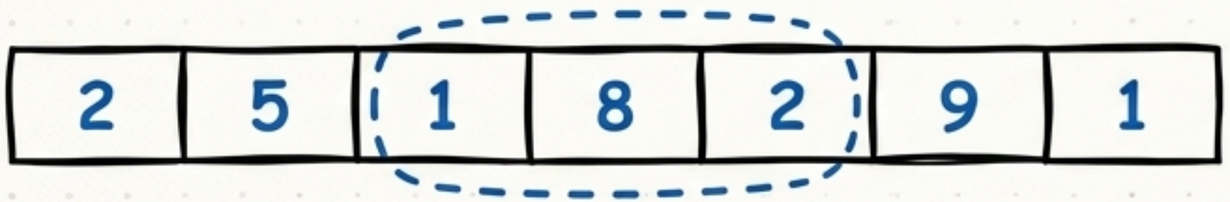


Sum = 8, Max = 8



Max = 14

Previous Sum (8) - Piche ka 2 cut + Aage ka 8 add = 14.



Max = 14 (No change)

Previous Sum (14) - Piche ka 5 cut + Aage ka 2 add = 11.

Dekho kaise ek element pichhe se cut raha hai, aur ek aage se jud raha hai without loop inside loop!

Clean Code - Max Sum Subarray 💻

```
int maxSum = 0, sum = 0;
int i = 0, j = 0;
while (j < arr.size()) {
    sum = sum + arr[j];
    if (j - i + 1 < k) {
        j++;
    } else if (j - i + 1 == k) {
        maxSum = max(maxSum, sum);
        sum = sum - arr[i];
        i++; j++;
    }
}
```

j wale ko add karo

Jab tak size K na ho, aage badho

Answer store

i ko remove karo

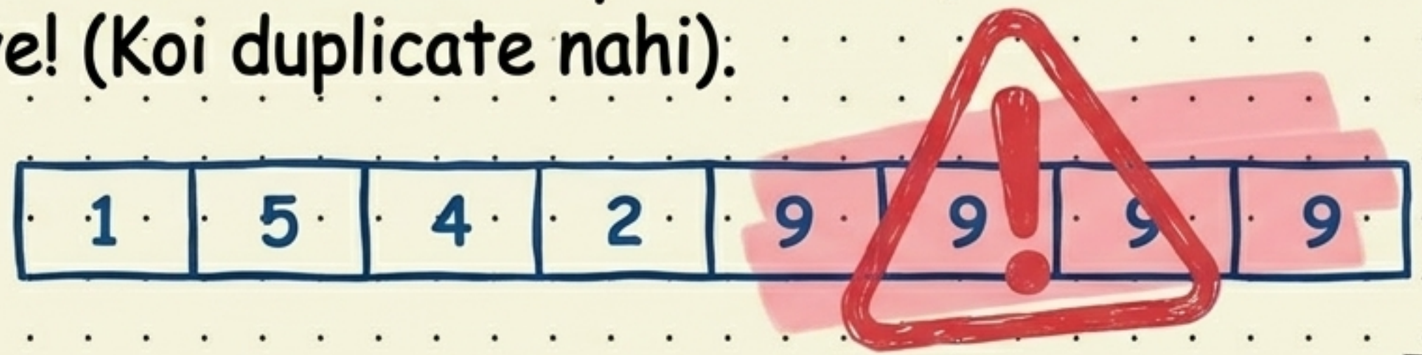
Window slide karo!

Time = O(N) (Kyunki pura array sirf ek baar traverse kiya)

Space = O(1) (Sirf pointers use kiye, koi extra memory nahi)

Twist! Leetcode 2461 (Distinct Elements) 🌪️

Problem: Max sum of subarray of size K, BUT saare elements distinct hone chahiye! (Koi duplicate nahi).

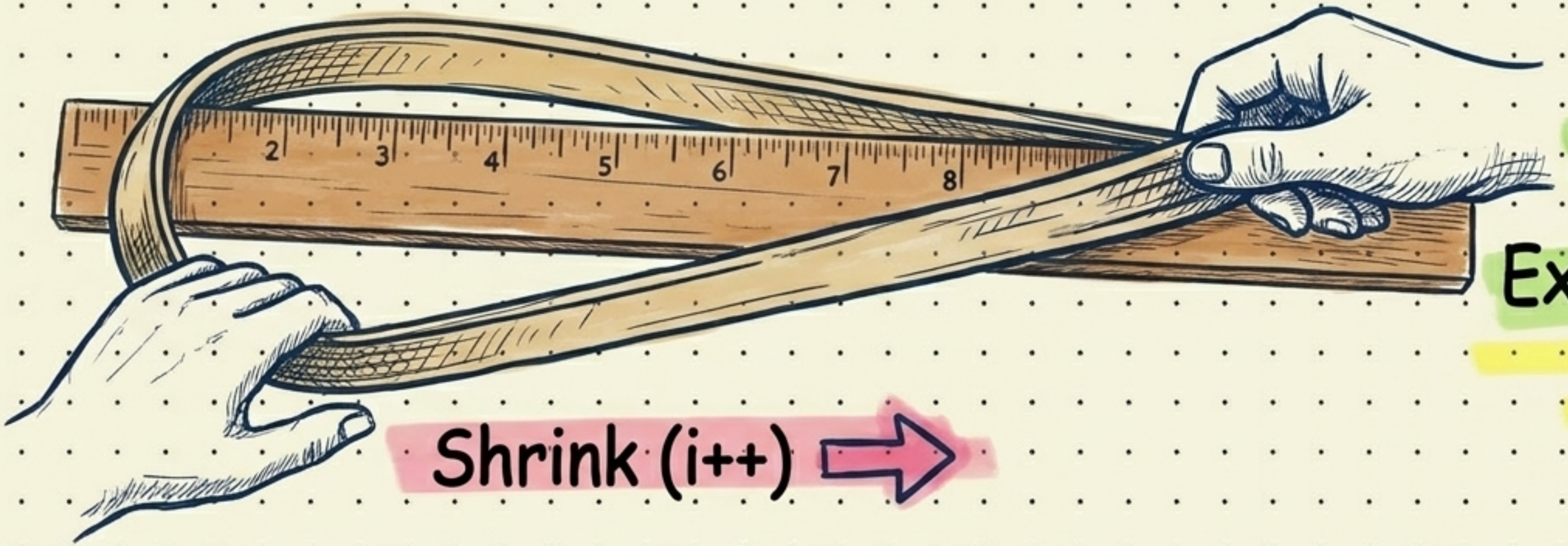


The Twist: Pattern wahi hai (Fixed Size), bas duplicates check karne hain.

Solution Logic: HashMap ya HashSet ka use karenge frequency count karne ke liye.

Rule: Agar Map/Set ka size == K hai, iska matlab window mein saare elements unique hain! Tabhi sum ko maxSum ke liye consider karo.

Variable Size Window (The Rubber Band Analogy)



Expand (j++)

Shrink (i++)

When to use: Jab **K** nahi diya ho! Sirf ek Condition di hoti hai (e.g., Sum ≥= Target).

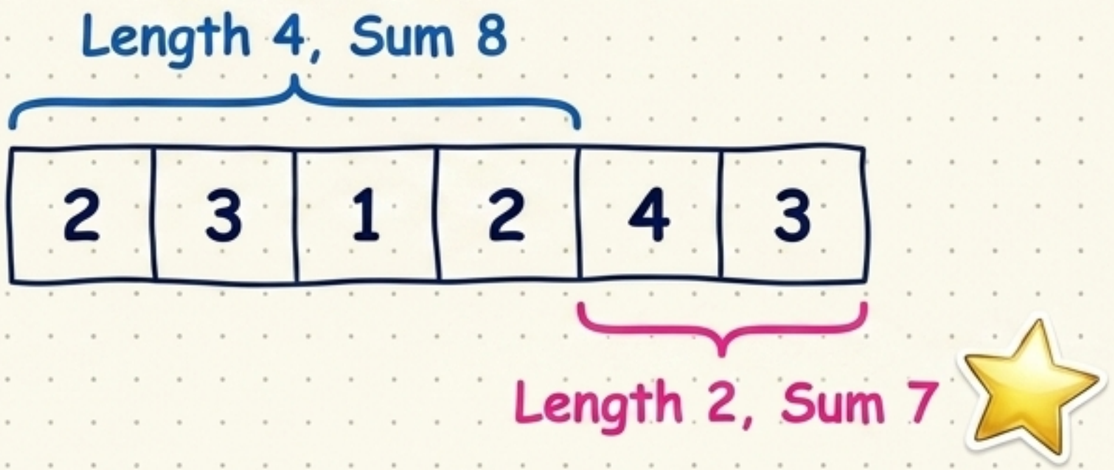
The Logic:

1. Expand: j ko aage badhao aur elements add karo jab tak condition meet na ho jaye.
2. Shrink: Jaise hi condition meet ho, i ko aage badha kar window choti karo (elements remove karo) aur answer check karo!

Problem 2 - Leetcode 209 (Min Size Subarray Sum) 🏆

Problem: Ek positive integers ka array aur ek target diya hai. Minimum length ka subarray find karo jiska sum >= target ho.

Target = 7



Identify the Pattern:

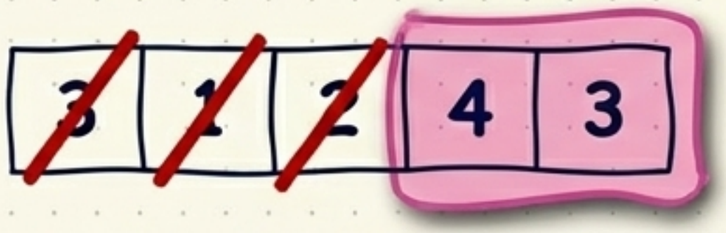
- Kya **Size K** diya hai? -> **NO.** ✗
- Kya **Sum** ki condition hit karni hai? -> **YES.** ✓

Conclusion: Yeh Variable Window hai! Length alag-alag ho sakti hai.

Goal: Hum chahte hain **sum >= 7** ho, aur window ka size sabse chota ho!

Dry Run – Expanding & Shrinking 🔍

Target = 7

Step 1 (Expand): Add elements using j. Sum = 8. (Valid! Length = 4).	
Step 2 (Shrink): Condition met! Try to make it smaller. Remove left 2 using i. Sum = 6. (Invalid now, but best length is still 4).	
Step 3 (Expand): Add next element using j. Sum = 10. (Valid! Length = 4).	
Step 4 (Shrink!): Remove left elements. Remove 3, then 1. Window becomes [4, 3]. Sum = 7. (Valid! Minimum Length = 2). 🎉	

Key Rule: j bada karega window, aur i chota karke answer ko optimize karega!

Clean Code - Leetcode 209 💻

```
int minLen = INT_MAX, sum = 0;
int i = 0, j = 0;

while (j < nums.size()) {
    sum += nums[j];  // Expand the window
    while (sum >= target) {
        minLen = min(minLen, j - i + 1);
        sum -= nums[i];  // Remove left element
        i++;  // Move left pointer
    }
    j++;
}
```

Condition met! Shrink it!

Myth Buster Box

Wait... Nested loops matlab $O(N^2)$ hona chahiye na? 🤔

NO!

Har element poore process mein exactly 1 baar add hota hai (j se) aur maximum 1 baar remove hota hai (i se).

Total operations = 2N. So Time Complexity = O(N) hi rahega! ⭐

Topper's DSA Notebook

Galtiyan jo Interview mein Hoti Hain 🛑

- ❌ Brute Force Trap: Question mein Continuous Subarray padh kar bhi $O(N^2)$ loop laga dena.
- ❌ Off-by-one Error: Fixed Window mein j - i + 1 (Size) ki condition theek se na lagana (size K ki jagah K-1 ya K+1 check karna).
- ❌ If vs While: Variable Window mein andar waale loop mein while ki jagah if lagana. (Humein tab tak shrink karna hai jab tak condition false na ho jaye!).
- ✅ Pro-Tip: Edge cases check karna mat bhoolna. Kya hoga agar poore array ka sum bhi target se chota ho? (Return 0).

Topper's DSA Notebook

1-Minute Revision Master Sheet ⚡

Fixed Window 📏

🔎 **Identify:** K (Size) is given.

➡ **Core Logic:** Expand till K. Then Slide by 1 (i++, j++).

{ **Condition:** if (j - i + 1 == K) }

Variable Window 🔁

❓ **Identify:** Condition is given (e.g., Sum > Target). Optimize for length.

➡ **Core Logic:** Expand j till condition meets. Shrink i aggressively while condition is met.

{ **Condition:** while (sum >= target) }

Pattern recognized. Repetitive work eliminated.
You are now a Sliding Window Pro! 💻🔥