Kadane's Algorithm

Monday, January 5, 2026 8:23 PM

1. The Goal

Find the contiguous subarray within a list of numbers that produces the largest sum.

2. The Core Intution

- "Don't carry dead weight."
- If a subarray sum becomes negative, it is useless for the future. Why? Because adding a negative number to the next number will always result in a smaller total than just starting fresh.
- The Rule: If your current running sum drops below 0, reset it to 0 immediately.

3. The logic flow (Pseudocode)

We iterate through the array once. At every number, we do three things:

1. ADD: Add the current number to "current_sum."
2. CHECK RECORD: Is "current_sum" > "max_sum"? If yes, update "max_sum".
3. RESET: Is "current_sum" < 0? If yes, reset "current_sum" to 0.

4. Visual Trace

Input: [-2, 1, -3, 4, -1, 2, 1, -5, 4]

Step	Number	current_sum	max_sum	Notes
0		0	0	Start
1	-2	-2 → 0	0	Neg? Reset to 0
2	1	1	1	New Record!
3	-3	-2 → 0	1	Neg? Reset to 0
4	4	4	4	New Record!
5	-1	3	4	Decreased, but still +
6	2	5	5	New Record!
7	1	6	6	Max found
8	-5	1	6	dropped, but > 0
9	4	5	6	End.

5. Complexity cheat sheet

- Time Complexity: O(n)
  - Why? We visit every number exactly once (linear time)
- Space Complexity: O(1)
  - Why? We only use two variables (max_sum and current_sum). We do not create a new array.