# Two Pointers – Core Patterns Summary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Pattern | Core Idea | Mental Idea | Typical Scenarios |
| 1 | Opposite Ends | Two pointers start at opposite ends and move toward each other until they meet. | The array is sorted or partially ordered — I’ll move inward based on a condition (sum too high → left, too low → right). | Pair/triplet sums, min/max area, palindrome check |
| 2 | Same Direction Sweep | Both pointers move forward, maintaining a relative distance or condition. | I’m scanning forward; left marks the start of a valid segment, right explores ahead. | Removing duplicates, moving zeros, maintaining a valid range |
| 3 | Fast–Slow Pointer (Runner Technique) | One pointer moves faster than the other to detect patterns, distances, or cycles. | Let one pointer explore ahead while another lags to compare positions or detect loops. | Linked list cycles, finding middle node, rearranging |
| 4 | Sliding Window (Dynamic Range) | One pointer expands the window, the other contracts it to maintain validity. | Grow the window until invalid, then shrink it until valid again. | Longest substring without repeats, minimum window substring |
| 5 | Partitioning (In-place Segregation) | Pointers move through the array to reorder elements based on a rule. | Swap elements so that one side meets a condition (e.g., < pivot). | Dutch National Flag, Move Zeroes, Partition Array by Parity |
| 6 | Merging (Sorted Merge) | Pointers iterate through two sorted sequences, merging them efficiently. | Advance the pointer that has the smaller current value. | Merge Sorted Arrays/Lists, Sorted Merge in mergesort |
| 7 | Meet-in-the-Middle | Two independent scans (sometimes on different arrays) that converge to satisfy a combined condition. | I’ll pick one element from each side and adjust until the joint condition fits. | Two-array sum problems, closest pair across arrays |
| 8 | Boundary Shrinking (Range Reduction) | Move boundaries (start/end) based on a monotonic property or constraint to minimize/maximize something. | Adjust range limits while the condition improves; stop when it no longer does. | Minimum size subarray sum, smallest/largest window problems |