

Two Sum

Solving the Two Sum problem - a classic hashmap based question with an efficient linear-time solution.

Strategy: Use a Hash Map:

1. Traverse the array once.
2. For each $\text{nums}[i]$, compute the complement: $\text{complement} = \text{target} - \text{nums}[i]$
3. Check if the complement already exists in the map:
 - If yes, return the indices: $[\text{map}[\text{complement}], i]$
 - Otherwise, store the current value in the map: $\text{map}[\text{nums}[i]] = i$

Why it works?: You're essentially asking: "Have I already seen a number that would add with $\text{nums}[i]$ to get target?"

Time and Space complexity:

- Time: $O(n)$: - each element is processed once.
- Space: $O(n)$: - worst-case storage in the map.

Edge Cases Handled:

- Duplicate values (like $[3, 3]$)
- Negative numbers
- Exactly one valid solution, as guaranteed by the problem.