Transparency Statement:

I learned to use hash maps.

I solved this with nested for loops resulting in a time complexity of O(N^2). In order to solve this with a time complexity of O(N) I asked ChatGPT what strategies I can use. It was only then where I learned how to use hash maps to reference data I have already visited

---

Prompt:

dont solve it for me but what methods would I use to change this solution from O(n^2) to something less time complex```

Response:

Great question — here are the methods and ideas you'd use to improve the time complexity:

1. Use a Hash Map (or Hash Set)
Instead of linearly searching the vector (find), store elements in a hash map for O(1) lookups.
Why it helps: Lookup in a hash map is on average O(1) vs. O(n) for linear search.
Use case: Common in problems like "two sum".

2. Preprocessing the Data

- Build a data structure (like a map or set) ahead of time that allows faster queries.
- Store values as keys in a hash map with their indices or counts.
- Reduces repeated work.

3. Two-pointer technique (for sorted arrays)
If the array is sorted or can be sorted, the two-pointer method helps in some subtraction-based problems.
Why it helps: After sorting (O(n log n)), you can solve the problem in O(n) time.
Caveat: Modifies order and changes problem semantics if indices matter.

4. Avoid duplicate work
If you're checking the same pairs multiple times, track which values you've already processed.

If you want a hint tailored to a specific problem (like "two sum"), let me know — I'll keep it abstract until then.