Flashcards – Hashmap Interview Practice

Each card presents a **question**. Try answering before flipping.

Card 1

Q: What data structure helps check if a character exists in another string in O(1) time?
A: `Set` – for membership tests like in Jewels and Stones.

Card 2

Q: How do you check if two strings are anagrams using a hashmap?
A: Compare their `Counter()` frequency maps: `Counter(s) == Counter(t)`.

Card 3

Q: Which problem asks if a word can be formed using available letters?
A: Ransom Note.

Card 4

Q: What is the key insight to group anagrams efficiently?
A: Use the **sorted word** as the key in a hashmap.

Card 5

Q: How do you solve Two Sum in O(n) time?
A: Store complements in a hashmap and check if current number exists.

Card 6

Q: How to validate uniqueness in rows, columns, and boxes in Sudoku?
A: Use a `set` with custom keys like `(i, num)`, `(num, j)`, `(i//3, j//3, num)`.

Card 7

Q: In Majority Element, what must the count of a number exceed?
A: `n // 2` (more than half of the array).

Card 8

Q: What does Longest Consecutive Sequence use to achieve O(n) time?
A: A `set` and only start counting if `n-1` doesn't exist.

Card 9

Q: How do you find how many times "balloon" can be formed from letters?
A: Use `Counter()` and divide counts for 'l' and 'o' by 2.

Card 10

Q: What's the fastest way to detect duplicates in a list?
A: Check `len(nums) != len(set(nums))`.

Quiz-Style Prompts

Try solving each of these mentally or in a scratchpad.

Q1:

Problem: Write a function that returns `True` if a string `s` can be constructed from another string `t`.
Expected Strategy: Use `Counter` and check `not (Counter(s) - Counter(t))`.

Q2:

Problem: You're given a list of integers. Find the number that occurs more than n//2 times.
Expected Strategy: Use a hashmap to count. Stop if `count[num] > n//2`.

Q3:

Problem: Group words like `["eat", "tea", "tan", "ate", "nat", "bat"]`.
Expected Strategy: `defaultdict(list)` with `sorted(word)` as key.

Q4:

Problem: How would you ensure a Sudoku board is valid in O(1) per cell?
Expected Strategy: Use 3 sets (rows, columns, boxes) to track uniqueness.

Q5:

Problem: Return True if a list has any duplicates.
Expected Strategy: Convert to set and compare lengths.

Advanced Round

Q6:

What data structure would you use to build an autocomplete system that suggests top words based on frequency?
A: A `Trie` with hashmaps at each level or hashmap of frequencies + heap.

Q7:

How can you optimize space if you know the input range is small?
A: Use arrays instead of hashmaps for frequency counts.

Want More?

- Convert these to Anki, Notion, or Quizlet cards
- Add custom difficulty levels (Easy, ⚙ Medium, Hard)
- Include coding timer challenges (e.g., solve in under 5 minutes)