Hashmap different dsa problem patterns, pattern details and most asked interview questions in each pattern

Sure! Here's a detailed breakdown of **HashMap-based problem patterns** in Data Structures & Algorithms (DSA), including common pattern details and frequently asked interview questions. This will help you recognize the problem type and approach solutions efficiently.

HashMap Problem Patterns in DSA

1. Frequency Counting / Element Counting

Pattern details:

- Use a HashMap to count occurrences of elements.
- Helps to answer queries like "how many times does X appear?", or check duplicates.
- Usually involves inserting elements as keys and frequencies as values.

Key operations:

- Increment frequency on insert.
- Check frequency for queries.

Common problems:

- Find the element with the maximum frequency.
- Check if two arrays have the same frequency distribution.
- Find if any element appears more than N/k times.
- Check if a string has all unique characters.

Interview questions:

- **Find the most frequent element in an array**
- **Check if two strings are anagrams**
- **Find duplicates in an array**
- **Longest substring with no repeating characters** (also a sliding window problem but frequency hashmap plays a key role)

---

2. Two Sum / Pair Sum Problems

Pattern details:

- Use HashMap to store elements and their indices.
- For each element `x`, check if `target - x` exists in the map.
- Key idea: Lookup in O(1) to find complements.

Key operations:

- Check if complement exists in map.
- Insert current element into map for future complements.

Common problems:

- Two sum variants.
- Pair with given sum in array.
- Count pairs with difference K.

Interview questions:

- **Two Sum (LeetCode 1)**
- **Count pairs with given sum**
- **Pair with difference K**

---

3. Subarray Sum Problems

Pattern details:

- Use a prefix sum + HashMap to store prefix sums and their indices.
- For target sum `k`, check if `prefix_sum - k` is present in the map.
- Helps in finding subarrays with given sum in O(n).

Key operations:

- Calculate running prefix sum.
- Store prefix sums with indices in map.
- Check for prefix_sum - k in map.

Common problems:

- Subarray with sum equals K.
- Count of subarrays with sum K.
- Longest subarray with sum K.

Interview questions:

- **Subarray Sum Equals K (LeetCode 560)**

- **Count subarrays with sum divisible by K**
- **Longest subarray with equal number of 0s and 1s**

---

4. Group Anagrams / Grouping by Key

Pattern details:

- Use a HashMap with a computed key (e.g., sorted string or frequency tuple).
- Group elements sharing the same key.

Key operations:

- Transform element into a canonical form (sorted string or frequency array).
- Store all elements with the same canonical form in a list in map.

Common problems:

- Group anagrams.
- Group strings with same characters.
- Group numbers by digit sums.

Interview questions:

- **Group Anagrams (LeetCode 49)**
- **Group shifted strings**

---

5. Longest Consecutive Sequence / Range Problems

Pattern details:

- Store elements in a HashSet or HashMap.
- For each element, check if element-1 is present to find start of a sequence.
- Extend sequence by checking element+1, element+2, etc.

Key operations:

- Check presence in map/set.
- Mark visited or skip already processed.

Common problems:

- Longest consecutive sequence of integers.
- Find longest subarray with consecutive numbers.

Interview questions:

- **Longest Consecutive Sequence (LeetCode 128)**
- **Longest consecutive subsequence**

---

6. Frequency Map + Sliding Window

Pattern details:

- Combine sliding window with hashmap frequency counts.
- Track frequencies of elements inside window to check conditions (e.g., all unique, count of distinct elements).

Key operations:

- Add/remove elements to/from frequency map when window slides.
- Check conditions based on frequency counts.

Common problems:

- Longest substring with at most K distinct characters.
- Smallest substring containing all chars of a pattern.
- Count of substrings with exactly K distinct chars.

Interview questions:

- **Minimum Window Substring (LeetCode 76)**
- **Longest substring with at most K distinct characters**
- **Find all anagrams in a string**

---

7. Counting Pairs with XOR or Other Bitwise Conditions

Pattern details:

- Use hashmap to store frequency of elements.
- Use properties of XOR or bitwise operators to find pairs quickly.

Key operations:

- Lookup complements based on XOR condition.
- Count pairs using frequency map.

Common problems:

- Count pairs with XOR = K.

- Subarrays with XOR = K.

Interview questions:

- **Count pairs with given XOR**
- **Subarray XOR equals K**

Summary Table

Pattern Type	Key Idea	Example Questions
Frequency Counting	Count occurrences	Find duplicates, max frequency
Two Sum / Pair Sum	Find complement in map	Two Sum, Pair with sum K
Subarray Sum	Prefix sums + hashmap lookup	Subarray sum = K
Grouping (Anagrams)	Canonical key to group	Group anagrams
Longest Consecutive Sequence	HashSet to detect sequences	Longest consecutive sequence
Sliding Window + Frequency Map	Frequency map inside window	Min window substring, Distinct chars
XOR / Bitwise Pair Counting	Map to count pairs by XOR condition	Pairs with XOR K

If you want, I can give you a detailed explanation + code for any of these patterns or specific problems! Would you like that?