Coding Important Topics

Important interview Questions

For a Lead Engineer role at Expedia Group, particularly with your experience in Java, Spring Boot, microservices, and Kafka, the interview is likely to cover a blend of **coding challenges** and **system design problems**. Below are some key areas and examples of interview questions that could help in preparation:

**1. Coding Questions**

The coding portion will likely focus on algorithms, data structures, and problem-solving in Java. Given that you're interviewing for a lead position, the problems may be a bit more complex, requiring optimization and clean coding practices.

**Common Algorithm Topics:**

* **Dynamic Programming**: Expect problems where you need to optimize solutions, such as subproblems being solved repeatedly.
* **Graph Algorithms**: Problems involving shortest paths, BFS, DFS, etc.
* **Greedy Algorithms**: Optimization problems.
* **String Manipulation**: Common coding challenge topic.
* **Trees and Graphs**: Traversals, path finding, etc.
* **Concurrency**: Thread-safe algorithms or producer-consumer problems can be asked, especially given your experience with Kafka.
* **Moore Voting** : algorithms

**Functional knapsack problem**

**Minimum operations to read at 10 from zero with two type of operations 1) plus1 or divide/2**

**Majority elements**

**Minimum platform**

**Sample Coding Questions:**

1. **Find All Anagrams in a String**:
   * Problem: Given a string s and a non-empty string p, find all the start indices of p's anagrams in s.
   * Approach: Use sliding window and hashmap to track counts.
2. **Merge Intervals**:
   * Problem: Given a collection of intervals, merge all overlapping intervals.
   * Approach: Sort intervals and then merge them in a single pass.
   * **Trains and Platforms**
3. **LRU Cache**:
   * Problem: Implement an LRU cache from scratch using Java's LinkedHashMap or Doubly Linked List + HashMap.
4. **Kth Largest Element in an Array**:
   * Problem: Given an unsorted array, find the kth largest element.
   * Approach: Use a priority queue (min-heap) or quickselect algorithm.
5. **Concurrency Problem (Producer-Consumer)**:
   * Problem: Implement a thread-safe producer-consumer pattern using BlockingQueue or synchronized.
   * You may also be asked to implement a ThreadPool in Java or deal with thread-safe collections.

**2. Design Questions**

Since the position is for a lead engineer, you will need to demonstrate proficiency in **low-level design** (OOP design, writing modular code) and **high-level architecture** (designing large-scale systems).

**Low-Level Design:**

Expect questions that test your ability to create clean, modular, and scalable object-oriented systems. Some examples:

1. **Design a Parking Lot System**:
   * Classes: ParkingLot, ParkingSpot, Vehicle, Ticket.
   * You need to consider capacity, types of parking spots, vehicle categories, and how to handle parking spot availability.
2. **Design an Elevator System**:
   * Classes: Elevator, Floor, Request.
   * You need to handle multiple elevators, requests for floors, and efficient movement between floors.

**High-Level System Design:**

You should be comfortable designing scalable, fault-tolerant systems. Questions will often involve designing systems like:

1. **Design an URL Shortening Service (like bit.ly)**:
   * Components: REST API, database for mapping short URLs to long URLs, hashing function for generating unique short URLs.
   * Consider scaling, database sharding, caching, and load balancing.
2. **Design an Online Booking System (similar to Expedia's core service)**:
   * Components: Microservices for handling search, booking, payments, inventory.
   * Focus on scalability, availability, database design (SQL or NoSQL), and consistency.
3. **Design a Rate Limiting System**:
   * Problem: How would you design a rate-limiting system for APIs to prevent misuse and DDOS attacks?
   * Techniques: Sliding window, token bucket, or leaky bucket algorithms.
4. **Design a Messaging Queue Service (similar to Kafka)**:
   * Components: Producers, consumers, brokers, partitions.
   * Focus on message durability, load balancing, and ensuring no message loss.
5. **Design a Distributed File Storage System (like Google Drive)**:
   * Components: Object storage, metadata service, replication, sharding.
   * Consider eventual consistency, high availability, and security.
6. **Design a Notification System**:
   * Components: User subscriptions, message delivery, push notifications.
   * Handle batching, prioritizing notifications, and dealing with high throughput.

**3. Microservices & Kafka Questions**

Given your experience in microservices and Kafka, the interview might explore your understanding of:

* **Microservices Architecture**:
  + How do you break down a monolithic application into microservices?
  + How do services communicate (synchronous via REST, asynchronous via messaging systems)?
  + Service discovery, load balancing, and circuit breaker patterns.
* **Event-Driven Architecture**:
  + How would you design a system using **Kafka** as an event bus?
  + How do you handle message partitioning and consumer groups?
  + How would you ensure that no events are lost, and how do you guarantee exactly-once delivery?
* **Data Consistency in Microservices**:
  + How would you ensure data consistency across distributed services?
  + Patterns like **Saga Pattern** for long-running transactions.

**4. Behavioral and Leadership Questions**

As a lead engineer, you'll also be assessed on your leadership qualities, teamwork, and how you mentor others. Some behavioral questions might include:

* How do you handle conflicts within a team?
* How do you manage deadlines and ensure the team delivers on time?
* Tell us about a challenging project you led and how you handled difficulties.
* How do you manage technical debt in a large-scale system?

**Suggested Preparation:**

1. **Coding**: Practice coding problems on platforms like **LeetCode**, **HackerRank**, and **GeeksforGeeks**. Focus on optimizing solutions and writing clean, modular code.
2. **System Design**: Read system design books like **Designing Data-Intensive Applications** by Martin Kleppmann and practice designing systems on **Pramp**, **Grokking the System Design Interview**, and other platforms.
3. **Java Microservices and Kafka**: Deep dive into best practices for building distributed systems using **Spring Boot** and **Kafka**, focusing on scaling, message handling, and fault tolerance.

Good luck with your interview at Expedia Group! Let me know if you need help with any specific concepts or topics.

For your upcoming interview at Expedia Group for the Lead Engineer role, here are some relevant coding and design topics frequently asked in 2024:

1. **Data Structures and Algorithms**: Expect questions on medium-to-hard Leetcode problems. Topics often include dynamic programming, two-pointer techniques, and arrays. Common examples include finding the longest increasing subsequence or sorting algorithms.
2. **System Design**: You might be asked to design large-scale systems like a booking or recommendation engine. Focus on scalability, high availability, and distributed systems.
3. **Low-Level Design (LLD)**: Object-oriented design questions related to real-world applications like designing an LRU cache or API frameworks.
4. **Behavioral & Leadership**: Emphasize team management, handling conflicts, and cross-functional collaboration, as well as specific challenges you've solved.

For technical rounds, focus on solving problems efficiently, discussing trade-offs in design decisions, and demonstrating leadership qualities.

4o

**You said:**

no, I am asking to list some coding problems which is most frequenlt asked in interviews or famous leetcode and hackerrank problems

**ChatGPT said:**

ChatGPT

Here are some frequently asked coding problems in interviews, as well as popular LeetCode and HackerRank problems:

1. **Two Sum** (LeetCode)
   * Problem: Find two numbers that add up to a specific target.
   * Type: Hashmaps, Arrays
2. **Longest Substring Without Repeating Characters** (LeetCode)
   * Problem: Find the longest substring without repeating characters.
   * Type: Sliding Window
3. **Merge Intervals** (LeetCode)
   * Problem: Merge overlapping intervals.
   * Type: Sorting, Greedy
4. **LRU Cache** (LeetCode)
   * Problem: Implement an LRU Cache.
   * Type: LinkedHashMap, HashMap
5. **Top K Frequent Elements** (LeetCode)
   * Problem: Find the k most frequent elements in an array.
   * Type: Heap, Sorting
6. **Product of Array Except Self** (LeetCode)
   * Problem: Return an array where each element is the product of all the other elements.
   * Type: Arrays
7. **Kth Largest Element in an Array** (LeetCode)
   * Problem: Find the kth largest element.
   * Type: Heap, Quickselect
8. **Find the Duplicate Number** (LeetCode)
   * Problem: Find the duplicate number in an array.
   * Type: Two Pointers, Cycle Detection

These are common problems you can practice for interviews on platforms like LeetCode or HackerRank.

4o

**You said:**

list some more coding problems i know already those

**ChatGPT said:**

ChatGPT

Here are some additional coding problems that are frequently asked in interviews:

1. **Coin Change Problem** (LeetCode)
   * Problem: Find the minimum number of coins required to make up a given amount.
   * Type: Dynamic Programming
2. **Word Ladder** (LeetCode)
   * Problem: Find the shortest transformation sequence from start word to end word.
   * Type: BFS, Graphs
3. **Maximal Rectangle** (LeetCode)
   * Problem: Find the largest rectangle of 1s in a binary matrix.
   * Type: Dynamic Programming, Stack
4. **Trapping Rain Water** (LeetCode)
   * Problem: Calculate the water trapped after raining.
   * Type: Two Pointers, Arrays
5. **Decode Ways** (LeetCode)
   * Problem: Decode a string of numbers where each number corresponds to a letter.
   * Type: Dynamic Programming
6. **Edit Distance** (LeetCode)
   * Problem: Compute the minimum number of operations to convert one word to another.
   * Type: Dynamic Programming
7. **Binary Tree Zigzag Level Order Traversal** (LeetCode)
   * Problem: Traverse a binary tree in a zigzag level order.
   * Type: Tree, BFS
8. **Meeting Rooms II** (LeetCode)
   * Problem: Find the minimum number of meeting rooms required.
   * Type: Greedy, Intervals, Heap
9. **Maximum Subarray Sum with One Deletion** (LeetCode)
   * Problem: Find the maximum subarray sum with at most one deletion allowed.
   * Type: Dynamic Programming
10. **Alien Dictionary** (LeetCode)
    * Problem: Determine the order of characters in an alien language.
    * Type: Topological Sort, Graphs

Practicing these problems will help with common patterns and techniques seen in coding interviews.

4o

**You said:**

list some more problems in string and array topic

**ChatGPT said:**

ChatGPT

Here are more coding problems focused on **strings** and **arrays**:

**String Problems:**

1. **Longest Palindromic Substring** (LeetCode)
   * Problem: Find the longest palindromic substring in a given string.
   * Type: Dynamic Programming, Expand Around Center
2. **Group Anagrams** (LeetCode)
   * Problem: Group strings that are anagrams of each other.
   * Type: Hashing, Sorting
3. **Minimum Window Substring** (LeetCode)
   * Problem: Find the minimum window in s that contains all characters of t.
   * Type: Sliding Window
4. **Valid Parentheses** (LeetCode)
   * Problem: Check if a string has valid parentheses.
   * Type: Stack
5. **Roman to Integer** (LeetCode)
   * Problem: Convert a Roman numeral string to an integer.
   * Type: String Manipulation
6. **Longest Common Prefix** (LeetCode)
   * Problem: Find the longest common prefix among an array of strings.
   * Type: String, Trie
7. **Substring with Concatenation of All Words** (LeetCode)
   * Problem: Find all starting indices of substring(s) formed by concatenating all words in the list.
   * Type: Sliding Window, Hashmap
8. **Multiply Strings** (LeetCode)
   * Problem: Multiply two non-negative numbers represented as strings.
   * Type: String Manipulation, Math

**Array Problems:**

1. **4Sum** (LeetCode)
   * Problem: Find four numbers that sum to a target.
   * Type: Two Pointers, Sorting
2. **Find the Missing Number** (LeetCode)
   * Problem: Find the missing number in an array of size n containing numbers from 0 to n.
   * Type: XOR, Sum Formula
3. **Search in Rotated Sorted Array** (LeetCode)
   * Problem: Find an element in a rotated sorted array.
   * Type: Binary Search
4. **Maximum Product Subarray** (LeetCode)
   * Problem: Find the contiguous subarray with the largest product.
   * Type: Dynamic Programming
5. **Jump Game** (LeetCode)
   * Problem: Determine if you can reach the last index in an array with jumps.
   * Type: Greedy, Dynamic Programming
6. **Find Minimum in Rotated Sorted Array** (LeetCode)
   * Problem: Find the minimum element in a rotated sorted array.
   * Type: Binary Search
7. **Find All Duplicates in an Array** (LeetCode)
   * Problem: Find all elements that appear twice in an array of size n.
   * Type: In-place Array Manipulation
8. **Spiral Matrix** (LeetCode)
   * Problem: Return all elements of a matrix in spiral order.
   * Type: Array Traversal

Practicing these problems will sharpen your skills in strings and arrays, which are crucial for coding interviews.

4o

**You said:**

list some linked list and stack and queue concepts coding problems

**ChatGPT said:**

ChatGPT

Here are some common **Linked List**, **Stack**, and **Queue** coding problems:

**Linked List Problems:**

1. **Reverse Linked List** (LeetCode)
   * Problem: Reverse a singly linked list.
   * Type: Iterative/Recursive
2. **Merge Two Sorted Lists** (LeetCode)
   * Problem: Merge two sorted linked lists into one sorted list.
   * Type: Recursion, Linked List
3. **Detect Cycle in a Linked List** (LeetCode)
   * Problem: Determine if a linked list has a cycle.
   * Type: Two Pointers, Floyd’s Cycle Detection
4. **Intersection of Two Linked Lists** (LeetCode)
   * Problem: Find the node where two linked lists intersect.
   * Type: Two Pointers
5. **Remove Nth Node from End of List** (LeetCode)
   * Problem: Remove the nth node from the end of a linked list.
   * Type: Two Pointers

**Stack Problems:**

1. **Evaluate Reverse Polish Notation** (LeetCode)
   * Problem: Evaluate an expression in reverse Polish notation.
   * Type: Stack, Math
2. **Min Stack** (LeetCode)
   * Problem: Design a stack that supports push, pop, and retrieving the minimum element in constant time.
   * Type: Stack, Design
3. **Valid Parentheses** (LeetCode)
   * Problem: Check if a string containing only parentheses is valid.
   * Type: Stack
4. **Largest Rectangle in Histogram** (LeetCode)
   * Problem: Find the largest rectangle in a histogram.
   * Type: Stack, Monotonic Stack

**Queue Problems:**

1. **Sliding Window Maximum** (LeetCode)
   * Problem: Find the maximum value in every sliding window of size k.
   * Type: Deque (Double-ended Queue)
2. **Implement Queue using Stacks** (LeetCode)
   * Problem: Implement a queue using two stacks.
   * Type: Stack, Queue
3. **Design Circular Queue** (LeetCode)
   * Problem: Design a circular queue with operations like enqueue, dequeue, etc.
   * Type: Queue, Design
4. **Number of Islands** (LeetCode)
   * Problem: Find the number of islands in a grid.
   * Type: BFS, Queue

These problems will help you master linked lists, stacks, and queues, which are essential for technical interviews.

4o

**You said:**

list some tree and graph based coding problems and shortest path problems

**ChatGPT said:**

ChatGPT

Here are some **Tree**, **Graph**, and **Shortest Path** based coding problems:

**Tree Problems:**

1. **Binary Tree Inorder Traversal** (LeetCode)
   * Problem: Traverse a binary tree in an inorder sequence.
   * Type: DFS, Recursion
2. **Lowest Common Ancestor of a Binary Tree** (LeetCode)
   * Problem: Find the lowest common ancestor of two nodes in a binary tree.
   * Type: Recursion, Tree
3. **Serialize and Deserialize Binary Tree** (LeetCode)
   * Problem: Convert a binary tree to a string and vice versa.
   * Type: Tree, BFS/DFS
4. **Binary Tree Maximum Path Sum** (LeetCode)
   * Problem: Find the maximum path sum in a binary tree.
   * Type: DFS, Recursion
5. **Construct Binary Tree from Preorder and Inorder Traversal** (LeetCode)
   * Problem: Reconstruct a binary tree from preorder and inorder traversals.
   * Type: Recursion, Tree

**Graph Problems:**

1. **Course Schedule** (LeetCode)
   * Problem: Determine if it is possible to finish all courses given prerequisites.
   * Type: Topological Sort, Graph, DFS/BFS
2. **Clone Graph** (LeetCode)
   * Problem: Clone a graph with nodes connected by edges.
   * Type: BFS/DFS, Graph
3. **Graph Valid Tree** (LeetCode)
   * Problem: Check if the given graph forms a valid tree.
   * Type: Graph, Union-Find
4. **Number of Connected Components in an Undirected Graph** (LeetCode)
   * Problem: Count the number of connected components in an undirected graph.
   * Type: Graph, Union-Find

**Shortest Path Problems:**

1. **Dijkstra's Algorithm** (LeetCode-style)
   * Problem: Find the shortest path from a source to all vertices in a weighted graph.
   * Type: Greedy, Priority Queue
2. **Bellman-Ford Algorithm** (LeetCode-style)
   * Problem: Find the shortest paths from a source to all vertices in a graph with possible negative weights.
   * Type: Dynamic Programming
3. **Floyd-Warshall Algorithm** (LeetCode-style)
   * Problem: Find all-pairs shortest paths in a graph.
   * Type: Dynamic Programming
4. **Cheapest Flights Within K Stops** (LeetCode)
   * Problem: Find the cheapest flight with up to k stops between two cities.
   * Type: BFS, Shortest Path
5. **Network Delay Time** (LeetCode)
   * Problem: Calculate how long it will take for all nodes to receive a signal from a starting node.
   * Type: Dijkstra’s Algorithm, BFS

Practicing these will help you master tree and graph traversal, shortest paths, and related concepts