Infosys Technical & Behavioral Interview Practice Sheet - 2

Instructions for students

- Go Beyond the Surface: For technical questions, aim to discuss trade-offs, design considerations, and real-world implications, not just definitions.
- Showcase Critical Thinking: For behavioral questions, emphasize your problem-solving process, decision-making rationale, and the lessons learned.
- Be Prepared to Elaborate: Interviewers might ask follow-up questions to delve deeper into your answers.

I. Technical interview questions

Programming language (choose one: C++, Java, Python)

1. Discuss the memory management mechanisms in your chosen language (e.g., garbage collection in Java/Python, manual memory management in C++). What are the advantages and disadvantages of each approach?
2. Explain the concept of thread safety and how you would ensure it in a multi-threaded application using your preferred language.
3. Describe a complex design pattern you've used (e.g., Observer, Factory, Singleton). Explain its purpose, implementation, and a real-world scenario where it would be beneficial.
4. Write a program to implement a custom data structure (e.g., a simple hash map, a trie) and discuss its time and space complexity for various operations.
5. Discuss the differences between pass-by-value, pass-by-reference, and pass-by-pointer (where applicable in your language). When would you choose one over the others?

Data structures and algorithms

1. Design an algorithm to find the shortest path in a weighted graph. Discuss the pros and cons of different algorithms (e.g., Dijkstra's, A*) and when each would be appropriate.
2. Given a large dataset (e.g., millions of records), describe how you would efficiently search for specific information or identify patterns using appropriate data structures and algorithms.
3. Implement an algorithm to sort a linked list without converting it to an array. Discuss the time and space complexity of your approach.
4. Explain how you would handle collisions in a hash map, discussing different techniques like separate chaining and open addressing.

5. Design a system to efficiently manage a dynamically growing dataset where frequent insertions, deletions, and searches are required. Choose appropriate data structures to optimize performance.

Database management systems (DBMS)

1. Explain the concept of database indexing and how it improves query performance. Discuss the different types of indexes and their trade-offs.
2. Describe a real-world scenario where you would choose a NoSQL database over a relational database, and vice versa. Justify your choice based on the project requirements.
3. Discuss the challenges and best practices for optimizing database performance in a production environment.
4. Explain transaction isolation levels in DBMS (e.g., Read Committed, Serializable) and their impact on concurrency and data consistency.
5. Design a database schema for a complex application (e.g., an e-commerce platform, a social media network). Discuss the design decisions, normalization levels, and potential scalability challenges.

Operating systems and computer networks

1. Explain the concept of deadlocks in an operating system. How can they be detected, prevented, and recovered from?
2. Discuss the differences between processes and threads, including their advantages and disadvantages in different scenarios.
3. Explain the concept of virtual memory in operating systems and its significance in modern computing.
4. Describe the various layers of the OSI model and their functions in network communication.
5. Discuss the security implications of different network protocols (e.g., HTTP vs. HTTPS, TCP vs. UDP).