**Generic Guidelines:**

1. Go through GeeksforGeeks and LeetCode web portals and familiarize with all as many as questions/solutions.
2. We’ve observed that presentation skills stand 50% weightage as much as 50% for technical. Some points to remember.
   1. It could be an exciting/intense situation and sometimes uncomfortable for some candidates when an interviewer watches him/her write live code, but haste will not help. Think of it as solving a puzzle with a colleague and create an opportunity for rapport building with interviewer.
   2. When a coding/algorithm question is asked or presented on coderpad, always take time to read. Then acknowledge what was understood with the interviewer.
   3. Confirm with interviewer on what output is expected before jumping into the code even though it might have been a familiar problem.
      1. If the candidate knows the problem, best if they exhibit as if they are solving it for first time. Reason being, at time it might work against their favor. When they solve it too fast, we’ve seen GS presents a third or fourth question. Usual interview is for candidate to solve two questions.
   4. Always excuse when time is needed for thinking. GS is not expecting the candidate to think and write code at the same time. For example: say “hey would you mind if I take a few mins to gather my thoughts before coding on the screen?” It potentially communicates that the candidate is well thought through. Small thing but speaks loudly about presentation skills.
   5. Simplest of things:
      1. Clean, decent, and well-lit surrounding when on video. Best practice is to use a neutral/suitable zoom background. If Genpact employee – use a Genpact background preferably.
      2. Attire – properly groomed.
      3. Good internet connections.
      4. Avoid taking calls from public places or background noises. Rather it is best to conduct the interview at a time when place is perfect. Move the interview timing if needed.
3. Though a candidate may have prepared all the questions in all portals, best to go with an open mindset to expect any type of algorithm coding and deal with it accordingly.

**JAVA (Senior/Lead Developer)**

**Round 1: Coding Round (Usually Two interviewers)**

1. Basic Intro and past exp.
2. DS & Algo question
   1. Code was given we need to correct the code and run all the given test cases
   2. They will ask one DS question we need to write a workable work and all the test cases should be passed.
   3. Finding second smallest element in an array
   4. Finding first repeated element in an array

**Round 2: Technical Discussions (Usually Two interviewers with different folks from before)**

1. Intro and past exp. in details (like domain, tech stack, etc.)
2. Explain one of the past or recent project in details (overview, tech stack, HLD, what problems you have faced, and how you overcome those problems)
3. SQL- index, joins, DB queries
4. Java theoretical question (Memory management, oops principles, overloading/overriding, multithreading, exception handling, etc)
5. Questions about the technologies whatever you mentioned in my resume
   1. Spring & Spring boot
   2. Kafka
   3. Redis
   4. ELK stack
   5. Hibernate
   6. Design Patterns

**REACT (Senior/Lead Front-End Developer)**

**Round 1: Coding Round**

1. DOM manipulation using vanilla Js script (3 questions)
2. Simple Algorithm question:
   1. Finding second smallest element in an array
   2. Finding first repeated element in an array
   3. Implement the put and contains methods of a binary tree
   4. Similar to "High Five" in LeetCode, but just get the highest overall score for each student instead of the average of 5
   5. Implement currying function
   6. Write a function to implement: const sum = calculateSum (1)(2)(3)(4)......();

**Round 2: Technical Discussions**

1. General react concepts
2. State management in React
3. React hooks
4. React lifecycle methods
5. Advanced javascript concepts:
   1. JavaScript promises
   2. Clousers
   3. Performance improvement (I was asked to implement debouncing in JavaScript)
6. General CI/CD related questions
7. Define shadowing in typescript
8. Misc questions about Redux
9. Misc questions about scss
10. Suppose you had a list of size n, which contained unordered integers from 0 to n-1, with 1 set of duplicates (e.g. [2, 1, 0, 2] or [0, 3, 1, 1, 2]). How would you find the duplicate in O(nlogn) (I think?) time complexity (no code, just talk through the general strategy)?

**Round 3: More Technical Discussions (Sometimes there is a Round 3 that GS asks)**

1. Suppose you were tasked with making a very slow page perform faster. What steps would you take to find the issue(s)?
2. Are you familiar with any charting tools, such as D3/HighCharts?
3. Suppose you have a PDF document that you want to display on a page, but don't want to allow copying/pasting, or printing of the document (note: you can convert it to HTML). How would you accomplish this?
4. Question about html canvas
5. How would you go about structuring a new React project (e.g. folder structure)?