1. Can you explain the process of writing and implementing high-level test cases for software verification?

In software testing, a "low-level test case" focuses on detailed, granular aspects of a feature with specific input values and expected results, while a "high-level test case" provides a broader overview of functionality, outlining general scenarios without defining exact inputs or outputs, essentially acting as a more abstract test case

2. How do you ensure that low-level requirements (LLRs) meet the necessary checklists?

DO-178C High-Level Requirements are intended to specify functional and operational requirements along with HW/SW interface requirements. Low-level requirements are supposed to be so detailed that you can code from them without needing more information.

- Are the requirements clear and unambiguous? (Are all aspects of the requirement understandable and not subject to misinterpretation? Is the requirement free from indefinite pronouns (this, these) and ambiguous terms (e.g., “as appropriate,” “etc.,” “and/or,” “but not limited to”)?)
- Are the requirements concise and simple?
- Do the requirements express only one thought per requirement statement, a stand-alone statement as opposed to multiple requirements in a single statement, or a paragraph that contains both requirements and rationale?
- Does the requirement statement have one subject and one predicate?

3. What is your approach to writing test scripts in C? Can you share an example of a test script you’ve written?

CSIM

4. Have you used DOORS, RTC, or JIRA before? How have you managed requirements using these tools?

Yes to doors, no to Jira

5. If a test case fails unexpectedly, what steps would you take to analyze and resolve the issue?

When a test case fails unexpectedly, the key steps to analyze and resolve the issue are: carefully reviewing the failure message and logs, attempting to reproduce the issue manually, investigating potential causes like recent code changes or environmental factors, analyzing the test script itself, verifying the testing environment, and then collaborating with developers to identify the root cause and implement a fix; ensuring to document the issue clearly and retest once the fix is applied.

6. Imagine you are assigned to fix a legacy C codebase that lacks proper documentation. How would you approach this task?

Gradually document existing code:
Start by creating high-level overviews of the system's architecture and key components, and then gradually add detailed explanations for individual functions and modules.
Use code commenting effectively:
Add clear and concise comments within the code to explain complex logic or non-obvious decisions.
Consider automated documentation tools:
Explore tools that can automatically generate documentation based on code analysis, providing a starting point for further refinement.
Write unit tests:
Comprehensive unit tests can act as a form of living documentation, demonstrating how the code is intended to be used and helping to understand its behavior.
Refactor and restructure code:
As you gain a better understanding of the code, consider carefully restructuring parts of the system to improve readability and maintainability

I hope you're doing well!

Aversan Inc., a reputable engineering company specializing in aerospace and defense, has **two exciting opportunities** on the same program for highly skilled professionals:

1. **Remote Software Verification Engineer**
   - **Location:** Remote (Canada/US), Eastern or Central Time Zones
   - **Mode:** Open to Full-Time or Contract
   - **Interview:** Single round (Zoom/team)
   - **Team:** Help develop and test low-level requirements per DO-178C for developing safety-critical graphics drivers and software.
2. **Key Skills:**
   - Proficiency in C programming, code reviews, and unit testing
   - Experience with tools like DOORS, RTC, JIRA, and GIT
   - Strong understanding of software testing methodologies
3. **Code Reviewer (Pathway to DevOps Role)**
   - Focus on code reviews with the potential to transition into a DevOps position on the same program

Next Step:
As part of the process, I've attached a C test for you to complete and submit back to me. Should not take more than 25/30 minutes.

Looking forward to your response!