DEBUGGING :----→

Key Strategies for Debugging

- **Understand the Error Message:** Carefully read and dissect the error or warning message. Many error messages provide specific information about what went wrong and where, including the file name, line number, and character position.
- **Reproduce the Bug:** Before attempting a fix, reliably reproduce the error. This confirms you understand the conditions under which it occurs and allows you to verify the fix later.
- **Use Debugging Tools:** Leverage your environment's built-in debugger. Step through your code line by line, inspect variable values, and observe the program's flow to pinpoint the exact location and cause of the error.
- **Utilize print Statements (Logging):** Strategically insert print or log statements to output the values of variables and confirm the program's execution path at different stages. This helps track the state of your application.
- **Isolate the Problem:** Comment out or temporarily remove sections of code to determine which part is causing the issue. This helps narrow down the scope of the problem.
- **Consult Documentation and Resources:** When a solution is not immediately obvious, search for the error message or relevant keywords online. **Stack Overflow** and official documentation (like MDN Web Docs) are valuable resources with a vast pool of solved problems and community knowledge.
- **Explain Your Code (Rubber Duck Debugging):** The act of explaining your code and the problem out loud to another person (or even an inanimate object like a rubber duck) can often lead to a breakthrough, as it forces you to articulate your assumptions and logic clearly.
- **Write Tests:** Implement unit tests to cover expected behaviors of your code. Writing tests helps identify existing bugs and prevents new ones (regressions) from being introduced by future changes.
- **Keep Code Simple and Readable:** Writing clean, well-structured, and readable code with clear function names and comments makes it easier to debug when errors inevitably arise.
- **Take a Break:** When frustrated or exhausted, step away from the code. A fresh perspective can often reveal a simple, obvious error that was missed.