Deliverable 5

A. Process Used to Find Errors

We followed a structured debugging approach to identify errors in the code:

1. Breakpoints in Chrome DevTools: Placed breakpoints at key function calls to inspect variable values.
2. Call Stack Analysis: Examined function calls to track how data flowed through the program.
3. Watch Expressions: Added key variables to the watch panel to monitor changes during execution.
4. Console Logging: Inserted console.log() statements in critical locations to verify intermediate values.
5. Step Over & Step Into Debugging: Used step-wise execution to observe where incorrect calculations occurred.

B. Differences Between Specifications and Implementation

- The specifications outlined correct tax calculations, but implementation errors caused miscalculations.
- Deduction handling in state tax computation was incorrect, adding instead of subtracting.
- Incorrect function usage (Math.dmax instead of Math.max) led to runtime errors.
- Federal tax bracket calculations had incorrect taxable income calculations, affecting results.

C. Suggested Improvements to Specifications

1. Provide Example Calculations: Including example inputs and expected outputs can help verify correctness.
2. Clearly Define Formulae: All formulas for tax calculations should be explicitly stated to avoid ambiguity.
3. List Data Type Requirements: Mention expected input types (e.g., numbers, valid tax brackets).
4. Include Error Handling Instructions: Specifications should specify how errors (e.g., missing brackets) should be handled.