Week 9: Final Testing & Debugging

Objective:

The primary goal of this week is to ensure the emulator is fully functional, efficient, and accurate by thoroughly testing it with various assembly programs, debugging any issues, and validating its performance against established benchmarks.

Tasks:

1. Test with a Variety of Assembly Programs

- **Purpose**: To ensure the emulator can handle different types of assembly code and edge cases.
- **Process**:
    - Create or gather a diverse set of assembly programs, including:
        - Simple arithmetic operations (e.g., addition, subtraction).
        - Control flow instructions (e.g., loops, conditional branches).
        - Memory access operations (e.g., load/store instructions).
        - Complex programs (e.g., sorting algorithms, recursive functions).
    - Run each program on the emulator and verify the output matches the expected results.
    - Test edge cases, such as:
        - Programs with invalid instructions.
        - Programs that exceed memory limits.
        - Programs with infinite loops or unexpected termination.
- **Outcome**: A comprehensive report detailing the results of each test case, including any discrepancies or failures.

---

2. Debug and Fix Any Issues

- **Purpose**: To identify and resolve any bugs or unexpected behavior in the emulator.
- **Process**:
    - Analyze the test results from Task 1 to identify issues.
    - Use debugging tools (e.g., logs, breakpoints, step-by-step execution) to trace the root cause of each issue.
    - Fix bugs related to:
        - Incorrect instruction execution.
        - Memory management errors.
        - Performance bottlenecks.
        - Unexpected crashes or hangs.

    - Re-test the emulator after each fix to ensure the issue is resolved and no new issues are introduced.
- **Outcome**: A log of all identified bugs, their root causes, and the fixes applied. The emulator should now handle all tested programs correctly.

---

3. Validate Performance Against Benchmarks

- **Purpose**: To ensure the emulator performs efficiently and meets performance expectations.
- **Process**:
    - Select a set of standardized benchmarks for emulators or similar systems.
    - Run the benchmarks on the emulator and measure key performance metrics, such as:
        - Execution speed (e.g., instructions per second).
        - Memory usage.
        - CPU utilization.
    - Compare the emulator's performance against:
        - Established benchmarks for similar systems.
        - Performance goals set during the design phase.
    - Optimize the emulator's code if performance falls short of expectations.
- **Outcome**: A performance report comparing the emulator's metrics to benchmarks, along with any optimizations made to improve performance.

Final Deliverables:

1. **Test Report**: A detailed document listing all tested assembly programs, their expected and actual outputs, and any issues encountered.
2. **Debug Log**: A log of all bugs identified, their root causes, and the fixes applied.
3. **Performance Report**: A comparison of the emulator's performance against benchmarks, including any optimizations made.
4. **Updated Emulator**: The final version of the emulator, fully tested, debugged, and optimized.

By the end of Week 9, the emulator should be robust, reliable, and ready for deployment or further development.