Defensive Programming Design Document

1. Overview

This document outlines the design of a robust, defensive programming approach for a Java application that checks for event overlaps based on user input. The main focus is on input validation, error handling, and maintaining system robustness.

2. Error-Handling Strategy

We classify errors into recoverable and fatal:
- **Recoverable errors:** Adjusting out-of-range values and swapping reversed date ranges.
- **Fatal errors:** Non-integer tokens, missing data, malformed input.
The program maintains correctness by halting on fatal errors and maintains robustness by correcting recoverable errors, logging appropriate warnings.

3. Local vs Global Error Handling

- **Local Handling:** Input parsing errors, such as format and range issues, are handled locally within the input validation module.
- **Global Handling:** Aggregated errors are reported globally to the user interface, which decides whether to proceed or terminate based on the presence of fatal errors.

4. Barricade Design

We implement a barricade pattern (input validation layer) to isolate untrusted data from internal logic:
- **Untrusted Zone:** User input (raw text lines).
- **Validation Layer (Barricade):** Parses, validates, corrects, or rejects inputs.
- **Trusted Zone:** Core logic assumes validated event data.

5. Defensive Programming Checklist Compliance

- **Protect from Bad Input:** Thorough input validation.
- **Assertions:** Used for conditions assumed to never occur post-validation.
- **Error-Handling Strategy:** Clearly defined and standardized.
- **Barricade:** Clear delineation between untrusted and trusted code.
- **Debugging Aids:** Meaningful error and warning messages.
- **Information Hiding:** Modular class design separates responsibilities.
- **Appropriate Defensive Code Amount:** Balanced validation without redundancy.

- **Exceptions:** Controlled use of exceptions for truly unexpected situations only.
- **Security:** Avoid sensitive information in error messages, ensure all errors are checked and handled.

6. Barricade Diagram