The Scenario: "Lehman Gradebook Processor"

The Registrar's office has provided a text file named `students.txt`. Each line contains a student's name and three test scores, separated by spaces. *Example:* `Alice_Smith 85 90 78`

Your Task

Create a Java program that reads this file and generates a summary report.

1. Setup the Files

- Manually create a file named `students.txt` in your project folder with at least 5 lines of sample data.
- One of those lines should have a "mistake" (e.g., a String where a number should be: `Bob_Jones 70 Error 90`) to test your exception handling.

2. Requirements

- **The Processor Class**:
  - Use a `try-with-resources` block to open a `Scanner` for reading `students.txt`.
  - Use a `PrintWriter` to create a new file named `grades_report.txt`.
- **The Logic**:
  - Read the file line by line.
  - For each student, calculate their average score.
  - **Exception Handling**: Wrap the score parsing logic in a `try-catch`. If a line contains invalid data (like "Error"), catch the `InputMismatchException` or `NumberFormatException`, print a warning to the console, and skip that student.
  - Write the successful results to `grades_report.txt` in the format: `Student: [Name] | Average: [Value]`.
- **The Final Summary**:
  - At the end of the program, use a `finally` block (or outside the try-with-resources) to print "Processing Complete" to the console.

Bonus Challenge (Optional)

If a student's average is below 60, throw a custom checked exception named `LowGradeException` and catch it to mark that student with a "Warning" flag in your report.

Checklist for Success

- [ ] Did you handle the `FileNotFoundException`?
- [ ] Is your `PrintWriter` inside the `try` so it closes automatically?
- [ ] Does your program stay running even if one line in the input file is corrupted?
- [ ] Did you use `hasNextLine()` or `hasNext()` to avoid `NoSuchElementException`?