I worked on an assignment to process and analyze a TV shows dataset of considerable size in a CSV format. The program considers importing data, filtering the shows, cleaning the list and exporting task-specific output files.

Most Challenging Aspect

CSV parsing logic implementation was cumbersome and took me quite some time to master it. The parseCsvLine() method was especially useful, yet difficult to implement due to a complex regular expression that deals with various wrapper systems present in CSV. The general areas in which I experienced considerable inconvenience were related to the presence of double quotes around fields, the possible existence of commas within quote encapsulations, and the possibility of different types of errors during data parsing.

Surprisingly Easy Tasks

Because integration patterns are new to me and the streams are broad, I anticipated that working with Java’s Stream API and lambda expressions would be overly complicated. Surprisingly, these patterns proved simple and robust, allowing for impressive productivity. Because methods such as filter(), sorted(), and limit() can all be called in one function, doing fairly intricate data transformations became easy. With Stream operations, finding the highest-rated 50 shows by concentration rating or looking at the shows and filtering them by their production status was an effortless venture.

Unexpectedly Complex Challenge

Writing data to files is what I expected to be an easy task, sadly turned out to be more complicated than expected. Some of the specific challenges include how to handle file writing exceptions or file closing, or the number of generated output files.

If I were to start this assignment again, I would make several improvements:

1. I would start earlier this time, giving myself more time to refactor and optimize the code.
2. Implement more robust error logging instead of simple print statements.
3. Create a more modular design by separating data processing logic from file I/O operations.

This assignment was an excellent learning experience in Java programming, data processing, and software design. It highlighted the importance of careful planning, robust error handling, and leveraging Java's powerful Stream API.