Act19: File I/O with Exceptions

Today, we look at reading from, and writing to, files in Java. Since file input/output (I/O) can often lead to errors (e.g., file not found), we will also learn how to handle exceptions that may be ***thrown*** during these operations.

Content Learning Targets

After completing this activity, you should be able to say:

- I can read/write files in Java using `java.io` classes.
- I can handle exceptions that may occur during file I/O operations.

Process Skill Goals

During the activity, you should make progress toward:

- N/A

Facilitation Notes

First Hour: Model 1 introduces file I/O and exception handling in Java. Model 2 focuses on designing file formats and working with files.

Second Hour: Model 3 provides time for project work on Milestone 1.

Model 1 File Input/Output

See the `src/fileIO` and `sol/fileIOsol` folders for the starter/solution code for this live coding activity. See also the relevant slides.

Model 2 Designing Project Files

In the LevelIO activity, we saw one possible way to save and load data using text files. However, there are many possible file formats we could use to represent the same data. In this activity, we revisit our chess program (which showed off the power of inheritance) to explore different ways to design file formats for saving and loading chess positions.

1. Brainstorm with your final project team: What are some possible ways we could represent a chess position in a text file? Consider how to represent the pieces, their locations, and any other relevant information (e.g., whose turn it is).

Answers will vary. Possible ideas: Use a simple text format with one line per piece, including its type and position; use a grid representation with characters representing pieces; use a CSV format with rows and columns for the board; include metadata such as whose turn it is or castling rights.

2. Live coding activity: As a class, we will implement two different file formats for saving and loading chess positions. One format will be a simple text format, and the other will be a CSV format. We will compare the two approaches and discuss their pros and cons.

3. Brainstorm with your final project team: What file format(s) will work best for your final project? Focus on ease of implementation and human readability; we won't worry about time/memory efficiency in this course.

Model 3 Project Work Time

Work with your final project team toward Milestone 1. Your project mentor will reach out with feedback on your M0, so use this time to address any suggested changes and continue making progress on your project. Consider setting up your file I/O structure as part of M1.