Task 1. Development

Implement a comprehensive test suite for the `MovieManager` class. The test suite should cover the functionality described below, including positive cases, negative cases, and corner cases, when applicable. Test cases must follow the best practices taught in the course and make effective use of JUnit 5 features. Add all test cases to the `MovieManagerTest` class.

1. Implement a setup method to initialise MovieManager using the movies.json file located in the resources directory. This setup must be reused in all test cases. (1 point)
2. Write one or more test cases to verify the functionality of adding a movie. (1 point)
3. Write one or more test cases to verify the search of movies by genre. (1 point)
4. Write a parameterised test to check that searching by exact title returns the correct movie. Input data (ex. movies' titles) should be provided in CSV format. (1.5 points)
5. Write a parameterised test to ensure that providing invalid ranges (min < 0, max > 10, min > max) when searching by rating throws an exception. Use an external method to generate the test data using @MethodSource. (1.5 points)
6. Write a test for searching movies within a rating range. Use JUnit assumptions to ensure the test is only executed if the dataset contains at least 20 movies. (1 point)
7. Use a mock MovieLoader that returns an empty list of movies and verify that searching by director returns empty results accordingly. (1.5 points)
8. Use a mock MovieLoader that returns a list containing duplicate movies (movies with the same title and director). Verify that the method for finding duplicates correctly identifies them. (1.5 points)