# Assignment 2

Implement the next release of your term project. You will incorporate exception handling and file I/O. (Please see the other attachments for a worked example solution in the form of an Eclipse project image.)

Submit this completed Word document, observing and retaining the gray text. Your materials—in black 12-point Times New Roman—should not exceed 5 pages excluding references, figures, and appendices. Use the Appendix sections for additional material if you need to. These will be read only on an as-needed basis.

As you code, use JUnit tests—package-by-package, class-by-class, and method-by-method, except for trivial ones and ones requiring I/O. Use non-Junit classes for testing the latter, as in the worked example. Keep the evaluation criteria in mind, listed at the end.

Include a ReadMe file describing where to run the application from, and including notes as necessary (not more).

## 1.1 SUMMARY DESCRIPTION

One- or two-paragraph overall description of your whole proposed term project. Edit your last description to improve it. Term projects will incorporate most of the techniques discussed in the course. To incorporate them, we recognize that you may need to alter your project or even introduce an additional project.

This project is an educational tool, titled MusicTrivia, for those looking to test and improve their music theory knowledge. The application will randomly generate questions from among several question templates ranging from simple to complex with random, appropriate values inserted at key points in the templates so that the user gets practice manipulating the questions with many different value permutations. After the user has selected his or her answer from among multiple options, the application will offer feedback on the correctness of his or her answer.

The application will store the users scores from session to session so that the user can track her/her cumulative progress over time.

## 1.2 PROJECTED I/O EXAMPLE FROM PROJECTED COMPLETED PROJECT

Provide an example of projected *concrete* output for designated input. You will not be held to fulfilling exactly this—it is just explanatory at this point, to indicate where your project is going. We recognize that project direction and details will change as the term progress. You are free to edit your previous description.

System:

Hello, welcome to MusicTrivia! Please enter your username.

User:

user1

System:

Welcome, user1. Your current scores are:

SimpleIntervalUpMajor: 5,

SimpleIntervalDownMajor: 2

SimpleIntervals: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

What is the 5th note of the melodic minor scale whose 7th note is the fifth overtone above C1? Enter the corresponding letter choice or type Q to quit:

A: Eb B: C C: E D: D

**User:** B

**System:**

Incorrect answer. Explanation: The fifth overtone above C1 is G. The melodic minor scale where G is the 7th is Ab melodic minor. The fifth note of the Ab melodic minor scale is Eb. The correct answer is Eb.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**System:**

What is a major 3rd above Bb? Enter answer or type Q to quit.

**User:**

D

**System:**

Correct! The correct answer is D.

SimpleIntervals score: 1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**System:**

What is a minor 2nd above B? Enter answer or type Q to quit

**User:**

Q

**System:**

Thanks for playing! Updated scores:

SimpleIntervalUpMajor: 5,

SimpleIntervalDownMajor: 2,

SimpleIntervals: 1

## 1.3 REQUIREMENTS IMPLEMENTED IN THIS RELEASE

1.3.1 Create Simple Questions

The application will create simple questions based on two templates that each use the MajorScale class as the theoretical background for the question material.

1.3.2 Pick questions at random and quiz the user

The application will pick from among many instantiated Question subclasses at random, display the question to the user, receive the user’s answer, evaluate the correctness of the answer, and display a simple message to the user about whether or not they were correct. The instantiation of objects is done with random, appropriate values.

### 1.3.3 Keep track of the number of correct answers for each question for each user

### The application will keep a record of the number of correct answers for each type of question for each user. These data will be stored in a JSON file. The complete score breakdown will be displayed to the user upon initial entry into the program and after exiting and the scores will be updated after each correct answer.

### 1.3.4 Prompt the user to self-identify before beginning

### Upon entry into the application, the system will prompt the user to enter his/her user name. If the username is not found in the system’s JSON file storage, a new entry will be made and all scores will be set to 0.

### 1.3.5 An explanation of how to solve the question is provided with the answer

As part of the feedback to the user, the system will provide an explanation of the answer to help with learning the material.

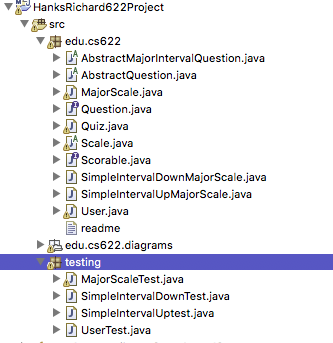
### 1.3.6 I/O Evidence that the Above Functionality was Achieved

Current contents of the JSON storage file

{"yjTiZAMclT":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"ExHZJLGWDe":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"GFcHAAEfOk":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"kBSMuCjuLk":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"EGjATPBZyh":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"bngGQFiTbl":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"lAxNqyTQHX":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"ZLxjZPXQaa":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"BPOpfDvxDe":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"cEsuCFMzXT":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"NfDYKAFFEm":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"TKHoUZkMDT":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"ZNFtyaTWgA":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"UsdWnnfjPD":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"tNzCskjxWa":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"iaokploIuD":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"GulKJoIiRd":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"MoRxWRdrnl":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"NdPXOMgZTh":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"xyIuvKSfFT":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"sJizCWyNnh":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"qzgEbfdylw":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"ytRGLowIEH":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"jISBEQxPjl":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"BuPJvPuVnA":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"dtKhdrQRgP":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"IeaFPiIvRZ":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"testUser2":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":3},"testUser1":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"QYeGurMxsF":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"categoryNames":["SimpleIntervalUpMajorScale","SimpleIntervalDownMajorScale"],"rozYMGbAZi":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"FBBGySAMYk":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"UYlSAfnRLr":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"AmSDZgVRjG":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"qqXFDjqBSd":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"testUser":{"SimpleIntervalDownMajorScale":8,"SimpleIntervalUpMajorScale":6},"HKZqWFztZQ":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"NpGQiJnNiL":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"UODfpXCYAH":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"ULLDvVAOrd":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"ABtYUirKrn":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"rmRKKZLbwq":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"WoaCDcSkMQ":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"ThCKQtONOw":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"HKyoEvdPVR":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"qFADDnygEF":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"lypZoDqvwV":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"CFsVsbTfHO":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"bVxsgIPToP":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"lsrvzYGwrd":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"fihROIDJtb":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"SQnMmNruwY":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"CSlgKjlFXK":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0},"rFrbJoYwJB":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":1},"XBnwkZRPTx":{"SimpleIntervalDownMajorScale":0,"SimpleIntervalUpMajorScale":0}}

## 1.4 YOUR DIRECTORY

Show a screenshot of your directory. This should include a parallel directory of JUnit tests—package-by-package, class-by-class, and method-by-method, except for trivial ones.



## 1.5 TECHNIQUES IMPLEMENTED

Your implementation should exploit file IO and exceptions at least once, in a natural manner. Explain where and how you applied these, using the headings below.

### 1.5.1 Class model and Sequence Diagram

Indicate clearly in your class model where you applied file IO and exception handling. Make classes and members *static* or not as per their intended usage. To do this use tools, PowerPoint, or a combine models as in [this example](https://docs.google.com/spreadsheets/d/1ZvkerE9FkWHWwVGdzuy7YMBU6oBMFGZbA4sotFETs8Y/edit?usp=sharing) (which you are free to cut and paste from). Insert indications in red (as in [this example](https://docs.google.com/spreadsheets/d/1ZvkerE9FkWHWwVGdzuy7YMBU6oBMFGZbA4sotFETs8Y/edit?usp=sharing)) to show where the three features below apply.

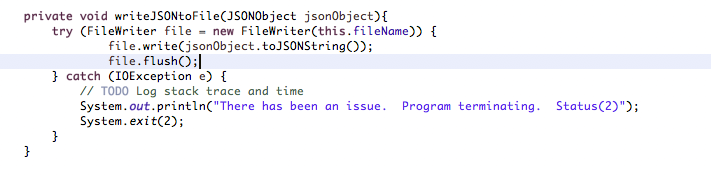


### 

### 1.5.2 Code showing File I/O

Show the relevant code (only). It should be clear where the code is located (class and method).

edu.cs622.User.writeJSONtoFile():

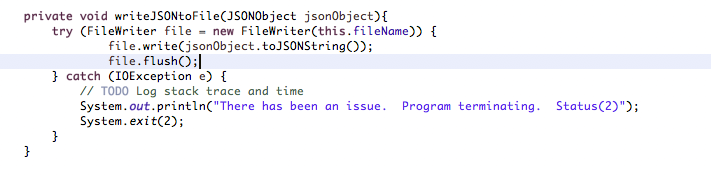


### 1.5.3 Code showing exception.

Show the relevant code (only) and explain why *exceptions* are appropriate here. It should be clear where the code is located (class and method).

edu.cs622.User..writeJSONtoFile():

Here I am catching an exception that is thrown if there is a problem with FileWriter writing to the file. In this case, because I don’t want the quiz to continue after this kind of error, I indicate to the user that the program is terminating and then I exit with an unique statue code for debugging purposes.

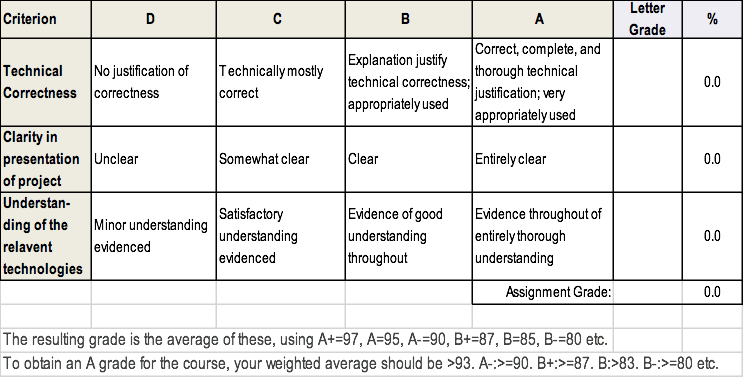


## 1.6 YOUR CODE

Unless your facilitator requests another method, copy your Eclipse project to your file system, zip it, and attach it. Please contact your facilitator in advance if you want to request an exception.

<Your response here>

## 1.7 Instructor’s Evaluation



## Appendix 1 (will be read as-needed only—add more as necessary)