Assignment: # 2 – Quiz Builder v2 (Individual)

Overview:
This Quiz Builder assignment is intended as an opportunity for students to demonstrate their knowledge of bundles, the hash and list data structures, file I/O, and parsing delimited text files.

Functionality:
This will be a two screen Android application that allows a user to prepare for 'matching-style' quizzes.

The first page of the application will present the user with a means of entering their name. The application will use an **intent** to navigate to the next page and pass this information from the first page of the application to the second page for display purposes. The second page will be where the testing itself takes place. The users score should be maintained and displayed on this page or via an intent on a third page once the testing is complete.

The second page (the quiz) will display one **definition** (i.e. a sentence/paragraph) in a large text box and four possible **terms** (i.e. single words), each on a button. The user will then select which term they think matches the definition displayed. The program will indicate if the user has made a correct or incorrect selection using colors, and or a message.

The GUI for this application is open to interpretation by you the developer, however, **storyboards** indicating UI components, layout and navigation **must be provided**, and your design should follow standard Android user interface design patterns and guidelines so that it is easy to use and visually appealing. If you wish to forgo a leading name page, then you must have a second page for the score that uses and intent/bundle combination.

Other Specifications:
- You may not 'hard code' the hash or lists with quiz data (i.e. terms /definitions) as this data must be read from a file.
- Include exactly ten term/definition pairs for testing and demonstration purposes.
- Provide validation for the input field (e.g. no empty strings)
- A definition may only be displayed once during a run of the program and then deleted.
- Terms should never be deleted as they are needed as a "pool" for other questions.
- None of the displayed terms (correct or incorrect) may be duplicates of each other.
- The correct answer should not always be in the same location from session to session
- Provided with the same quiz data in a file, the program should display the definitions in a different order from run to run. (Hint – 'shuffle')
- The quiz should end if it has run out of definitions to be displayed and the user should be asked if they want to do the quiz again.
- The program should use two Array Lists and one Hash. The lists to hold terms and definitions, with the Hash to hold their relationship to each other.
- The program must store its quiz data in a delimited text file
- Use Androids **logging mechanism** to log error messages via **try/catch statements**.
- Efficiency is very important – try not to create any more data structures (lists, hashes, arrays) or make more trips to these than necessary given the requirements.

Getting Started/Recommended Steps:

- Use a series of prototypes to examine and test the various concepts (e.g. file I/O, parsing, etc.). Doing so will help provide a starting point and focus developers on one technology at a time.
- Use the functionality built into data structures rather than write custom routines; where possible (e.g. 'shuffle', 'delete/remove')