CSCI 235 - Software Analysis and Design II
Project 01

This project will allow you to practice designing object-oriented software, practicing file I/O, using the String class, using command line arguments, using the gdb debugger, and learn how to create a makefile.

You task is to write a Word Jumble game. The game will jumble a word and allow the player to guess the original word. Below are the game specifications:

Game rules:

- The game should display a randomly jumbled word and allow the user to guess the original word.
- If the player's guess is incorrect, the player should be prompted to guess again.
- If the player's guess is incorrect 3 times for the same word, the game is over.
- If the player correctly guesses 3 words of the same difficulty level, the game should advance to the next difficult level (as described below under game logic).

LifeLines:

- The player may opt to switch out the jumbled word for a different word up to 5 times during the game.
- Using a lifeline can only be done before the third wrong guess, i.e. before the game is over.

Game data:

- The game should accept a text file as its first argument on command line.
- This file should contain the words to be used in the game (this data should be parsed as described below).
- If the game is invoked omitting the first argument, the game should load the previously parsed input data.

Parsing the input file:

- The game should be capable of parsing any text file.
- The input file should be parsed before the game starts.
- The words used in the game should only contain lower case letters, i.e. no uppercase letters, no numbers, no punctuation, etc.

- The parsed game data should not contain any duplicate words.
- The parsed data should not contain any words of length < 3.
- The parsed data should have at least 3 occurrences of words of the same length.

Game logic:

- The difficulty level of the game is determined by the word length. The longer the word the more difficult the game becomes.
- The game should start by displaying a word at the lowest difficulty level, i.e. the shortest word from the parsed game data (but >= 3 characters long).
- If the player's guess is correct 3 time for the same difficulty level, the game should advance to the next difficulty level, i.e. a longer words (the next length will be determined by the input data).
- The game should not display the same word twice during the duration of one game iteration.
- The game should keep track of the number of correct words guessed so far.
- The game should keep track of the longest word guessed so far, i.e. the difficult level reached.
- The game should allow the player to quit at any point in the game.
- If the player quits in the middle of a game, the current state of the game should be saved. Once the player loads the game again, there should be a prompt to allow the player to continue the previous game. Although, if an input file is specified on command line, the saved game data will be lost and new game data parsed.

Game statistics:

- When the game starts, it should display the most number of words the user guessed correctly and the difficulty level reached (the longest word guessed) across all games played for the current input data.
- Each time a new input file is parsed, the statistics should be reset.

It is up to you to design the user interface, but the game play should be intuitive and the following should be displayed:

- The best game statistics
- The current game statistics
- Number of lifelines left
- Option to *switch* words and use a lifeline
- Options to *quit* and *reset* game

Your should follow the OOAD approach as discussed in Chapter 1 when designing your code and writing your game. Make sure that you adhere to test-driven development (Appendix B) and use the gdb debugger.

Submission: Please upload the project to BB as a .zip file. You should include a README file describing your project — UI design, code design, how to compile the project (makefile) and how to use the game— and a makefile to allow the user to automatically compile the project. You can search online for makefile tutorials to learn more about how to write one.