FIT9131 Assignment A: Learndle

Introduction

This assignment is due by **11:55pm AEST on Friday of Week 7 (8 September, 2023)**. In addition there is a requirement that you show your work to your tutor in your Applied class in weeks 5 and 6, and attend an interview with your tutor following the submission of your assignment.

The assignment is worth 15% of the marks for your final assessment in this unit. **Heavy penalties will apply for late submission.** This is an **individual** assignment and must be your own work. You must attribute the source of any part of your code which you have not written yourself. Please note the section on plagiarism in this document.

You **must not use generative artificial intelligence (AI)** to generate any materials or content in relation to any assessment task in this unit.

The assignment must be done using the Assignment A workspace environment on the Ed platform (opposite). All code for the assignment must be written in this workspace and not cut and pasted from another place.

The Java source code for this assignment must be implemented according to the **FIT9131 Coding Standards** (see Lesson 3.4).

Any points needing clarification may be discussed with your tutor in the applied learning classes.

Completion of this assignment contributes towards the following FIT9131 learning outcomes:

1. Design and construct Java programs according to standard object-oriented principles

2. Apply and demonstrate debugging processes to Java applications

3. Develop strategies for efficient and effective program testing

4. Document code according to specific programming standards

Specification

For this assignment, you will write a program for a person to play a word game called *Learndle*. This section specifies the required functionality of the program. **Only a text interface is required for this program**; however, more marks will be gained for a game that is easy to follow with clear information/error messages to the player.

The aim of *Learndle* is for a player to correctly guess a hidden word in a set number of attempts. After each incorrect guess the player is shown the letters in the word they have guessed that are in the correct position in the hidden word. The game continues until the player has correctly guessed the word or they have run out of attempts.

The player may play a series of games. At the end of each game the player is awarded a score according to the number of attempts they have taken to guess the word. At the end of the series of games the player is shown their maximum game score.

Rules of the Learndle game

1. The *Learndle* game begins with a message inviting the player to enter their name. The name can contain only contain lower case alphabetic characters and must be no more than eight characters in length. If the player enters an invalid name then a warning message is displayed and the player is invited to enter another name. This will not count as an attempt.

2. The length of the hidden word that the player has to guess is displayed on the screen. The word length is randomly chosen to be either four, five or six characters. Each word length is equally possible.

3. The hidden word is chosen by the computer. The word is not shown to the player. To generate the hidden word, you'll need to use the provided `LearndleKeywords.class` file. This class contains a static method called `generateRandomKeyword(length)`, which generates a hidden word based on the specified length. The length should be between 4 and 6, inclusive - **LearndleKeywords.generateRandomKeyword(length).**

4. The maximum number of attempts allowed to guess a word is one less than the length of the word. For example, for a five-character word, four attempts are allowed.

5. The player is then invited to enter a word to guess the hidden word. The word entered must be the same length as the hidden word. If the word length does not match that of the hidden word then the guess is invalid. In this case the player is given a warning message and no information is given about the whether any characters are correct. This will not count as an attempt.

For each valid guess the following outcomes are possible:

-

The player has correctly guessed the word. In this case the game ends and the player is shown their game score (see rule 6).

-
-

The player has incorrectly guessed the word and has used all their attempts. In this case the game ends and the player is shown their game score (see rule 6).

-
-

The player has incorrectly guessed the word and has not used all their attempts. In this case the player is shown the correctly guessed letters for the hidden word. For each character position in the guessed word:

-
  -

If the guessed character matches the character at the same position in the hidden word, that character is displayed.

- ○
- ○

If the guessed character does not match, an underscore character (_) is displayed.

- ○

For example, if the hidden word is “number” and the player enters “queues” then the player would be shown “_u_ _e_”. This indicates that the second character in the hidden word is "u" and the fourth character is "e," while the other characters are still unknown. This feedback helps the player narrow down the possible correct letters in the hidden word and make more informed guesses in subsequent attempts. The game continues until the player either guesses the word correctly, runs out of attempts, or chooses to end the game.

6. At the end of the game the game score is displayed to the player. The game score is calculated as follows:

*game score = (word length - number of attempts) * correct guess*

where the *correct guess* is 0 (zero) if the correct word was not guessed and 10 if the correct word was guessed.

7. At the end of the game the player is then invited to play another game.

8. If the player decides to not continue with another game then their highest game score is displayed.

Program design

Your program should consist of at least three classes: Player, Learndle, and WordLength The following sections give details of these classes.

Player class

The Player class will specify the attributes and behaviours of a player. An object of the Player class will have at least the following fields:

Name – the name of the player

Score – the highest game score

The data type of each field must be chosen carefully and you must be able to justify the choice of the data type of the fields. You may want to include comments in the class to state any assumptions made. The class must also have a default constructor and a non-default constructor that accepts a value for the name of the player.

The Player class should also have *appropriate* accessor and mutator methods for its fields. Validation of values for fields should also be implemented. You should not allow an object of class Player to be set to an invalid state. There should be no input from the terminal or output to the screen via methods from the Player class. A Player object should also be able to return its state in the form of a String.

Learndle class

The Learndle class will manage the playing of a the game. It will have the following fields (at least):

Player - an object of type Player

Word length - the length of the hidden word

Hidden word - the word the player will attempt to guess

The Learndle class will have methods to manage the playing of the game. These should include the **main()** method to start the program and methods for the following behaviours (at least):

-

Display a welcome message on the screen.

-
-

Request the player to enter their name.

-
-

Get the length of the hidden word

-
-

Get the hidden word

-
-

Display the result of guess.

-
-

Display the score at the end the game.

-

- Display the highest game score

WordLength class

An object of the WordLength class will generate a random word length from a minimum to a maximum word length, inclusive.

Assessment

Assessment for this assignment will be done via an **interview** with your tutor. The marks will be allocated as follows:

- 10% - Progress of code development, as shown via Ed workspace environment. Your tutor will assess your work during your applied session in weeks 5 and 6.
  - 5% in **week 5** for Class Diagram (this can be hand-drawn). Note that the class diagram should show the individual classes and the interactions between the classes but does not need to include the details within the classes.
  - 5% in **week 6** for **Player** and **WordLength** classes
- 10% - Class diagram (including class interactions and internal details), Java code quality and object-oriented design quality. This will be assessed on code quality (e.g. compliance with coding standards) appropriate design and implementation of classes, fields, constructors, methods, and validation of the object’s state.
- 20% - Program functionality in accordance to the requirements.
-

-

60% - Oral assessment.

-

Assessment for this assignment will be done via an **interview** with your tutor. The marks will be allocated as follows:

Please note the following:

-

You must use the workspace environment in the Ed platform to code all parts of your program. You must not copy paste huge chunks of code from somewhere else.

-
-

You must ensure that your program code meets the coding standard requirements outlined in FIT9131 Java Coding Standards..

-

Marks will be deducted for any submissions that do not comply with the above.

You must submit your work by the submission deadline on the due date (a late penalty of 10% per day, inclusive of weekends, of the possible marks will apply). There will be no extensions - so start working on it early!

Interview

You will be asked to demonstrate your program at an interview following the submission date. At the interview, you will be asked to explain your code, your program design, modify your code, and discuss your design decisions and alternatives. Marks will not be awarded for any section of code/design/functionality that you cannot explain satisfactorily (the marker may also delete excessive in-code comments before you are asked to explain that code).

In other words, you will be assessed on your understanding of the code, and not on the actual code itself.

Interview times will be arranged in the applied classes in Week 7 and will take place on campus.

It is your responsibility to make yourself available for an interview time. **Any student who does not attend an interview will receive a fail grade for the assignment.**

Submission Requirements

The assignment must be submitted by **11:55pm AEST on Friday of Week 7 (8 September, 2023).**

The submission requirements for Assignment A are as follows:

-

The main class in your program **MUST** be called **Learndle.java** and it should contain the **main()** method to start the program.

-
-

Submit your work via the Ed platform, using the workspace area next to this assignment specification.

-
-

Re-submissions are allowed before the submission deadline, and are encouraged. You should submit your progress as you go. However, please ensure that you do not click on the submit button *after* the due date. The last submission will be used for grading purposes and a submission after the deadline will incur a late penalty.

-
-

A signed Assignment Cover Sheet. [Note: You are required to download the Assignment Coversheet, sign the document and upload the pdf file in the Ed platform (you may drag and drop to the Toggle Pane)]

-

Marks will be deducted for any of these requirements that are not complied with.

Warning: there will be no extensions to the due date. Any late submission will incur a 10% per day penalty. It is strongly suggested that you submit the assignment well before the deadline, in case there are some unexpected complications on the day (e.g. interruptions to your home internet connection).