University of St Andrews

School of Computer Science

CS5031 - Software Engineering Practice

Assignment: **P1 - Wordle**

Deadline: 13 February 2023 *Credits*: 20% of coursework mark

MMS is the definitive source for deadline and credit details

You are expected to have read and understood all the information in this specification and any accompanying documents at least a week before the deadline. You must contact the lecturer regarding any queries well in advance of the deadline.

Aim / Learning objectives

The aim of this assignment is to practise the use of test-driven development as well as good coding practices, refactoring methods, build tools, and version control.

Requirements

You are required to develop a portable Java version of Wordle, a word game where players have six attempts to guess a five-letter word.

The game must be developed using a test-driven development process and use Junit 5 as the unit-testing framework. The code must build, and tests must run using the Maven configuration file and commands.

You must use Git to version-control the code so that you can easily revert to a previous state if anything goes wrong, and the marker can assess how you have used a step-wise, systematic approach following the logic of test-driven development. For this purpose, check in versions of your code that pass the current set of unit tests before you add further tests and improve the quality of the code.

You must document all code with comments or JavaDoc. You may use Maven plugins to generate JavaDoc where appropriate; however, provide instructions on plugin usage in your report.

You can find the rules for the Wordle game at the end of this document. The code must be able to model the Wordle rules and game state. Your code must be able to be compiled into a portable jar file using Maven and run from the command line (as opposed to running from your IDE).

The code must select a word randomly from the provided wordlist and accept user input (guesses) from the command line with correct error handling.

After each input, the appropriate feedback should be displayed to the user before accepting another input. Finally, your code should provide a score to the user based on how many guesses were required.

You do not need to recreate the same user interface as the web-based Wordle. A simple command line interface is sufficient. To clarify, you may use coloured outputs like in the web-based Wordle or instead use non-alphabetical characters. For example, using * to represent a correct letter, ^ for the wrong position correct letter, and # for an incorrect letter.

You may add new features to your implementation that presently do not exist in the web-based Wordle.

Report

Your report must include:

- An overview of your Wordle implementation (How to Play)
- A description of your test-driven approach
- A description of your refactoring approach throughout development
- A description of any other software development practices used
- Any other relevant discussions, such as a description of extra functionality if provided
- References

The length of the report should be *no more than 800* words. This is an advisory limit.

What you are provided with

The zip file at https://studres.cs.st-andrews.ac.uk/CS5031/Practicals/pract1/wordle.zip includes a project you can import into your favourite IDE. You must initialise a git repository in the project's directory. The project contains two wordlists: `src/main/resources/wordlist.txt` and `src/test/resources/wordlist-test.txt`. `wordlist.txt` contains the entire list of valid five-letter words for Wordle; use it for playing your Wordle implementation. `wordlist-test.txt` contains three five-letter words to be used for your unit tests. You must use these as the wordlists. You must implement the model for the game using the contents of this project as a starting point and commit successive partial implementations to your git repository.

Autochecking

This assignment does not use stacscheck. You are expected to write your own tests.

Submission

Create a single zip file containing a Git repository containing:

(1) The incremental versions of the source code.
(2) A README file with instructions on building, unit testing, and running your code.
(3) A PDF copy of your report.

The zip file must be submitted electronically via MMS by the deadline. The latest version of the code should be checked out in the workspace. Submissions in any other format will be rejected.

Marking

Marking will follow the guidelines given in the school student handbook (http://info.cs.st-andrews.ac.uk/student-handbook/learning-teaching/feedback.html#Mark_Descriptors).

Some specific descriptors for this assignment are given below:

Mark range	Descriptor
1 - 6	An attempt to develop a Wordle model that fails to implement data structures and game logic for the game. A minimal attempt at a report showing a lack of understanding of the requirements.
7 - 10	Successful implementation of the game logic but with significant problems, either in terms of faults in the implementation or in not following a test-driven process with no or few unit tests produced. A reasonable attempt at a structured report with some evidence covering the required information.
11 - 13	A version of the game that may have shortcomings in the quality of the implementation or may not pass all the unit tests. Evidence of an attempt to follow the test-driven development process. A competent attempt at the report with a clear structure and writing covering most required aspects.
14 - 16	An implementation that passes all the unit tests. Unit tests cover all functionality, including edge cases and executing all branches in the code. The code quality might need to be revised. Evidence that the test-driven development process was used. Good attempt at the report addressing all required aspects in good writing without major problems.
17 - 20	An implementation that is fully working and thoroughly tested, applying the test-driven approach and evidence of code quality refinements through refactoring methods, leading to excellent code quality. A well-written report outlining excellent work addressing all required aspects.

Lateness

The standard penalty for late submission applies (Scheme B: 1 mark per 8 hour period, or part thereof): http://info.cs.st-andrews.ac.uk/student-handbook/learningteaching/assessment.html#lateness-penalties

Good Academic Practice

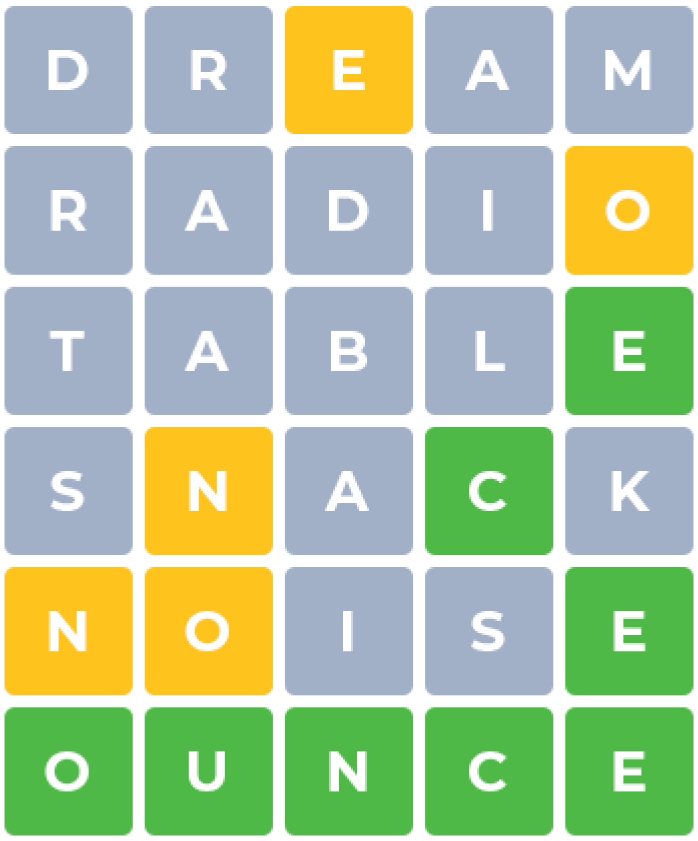
The University policy on Good Academic Practice applies: https://www.st-andrews.ac.uk/students/rules/academicpractice/

About Wordle

The objective of the game Wordle is to guess the correct five-letter word in six or fewer tries. The five-letter word is unknown to the player until a correct guess, or there are no more guesses. After each guess, feedback is provided depending on the position and correctness of each letter.

- A correct letter in the correct place turns green.
- A correct letter in the wrong place turns yellow.
- An incorrect letter turns grey.

In addition, letters can be used more than once and guessed words must exist in the wordlist (Note: the wordlist does not contain all five-letter words).



An example game of Wordle

You can try out the original web-based version of Wordle at https://www.nytimes.com/games/wordle/ to get a feel for how it is played.