

Advanced Object Oriented Programming

Surname:	Yiting
First Name:	Wang
Student Number:	201918020103
Module Code:	CHC 6186
Date Submitted:	May 22, 2023

1. Design

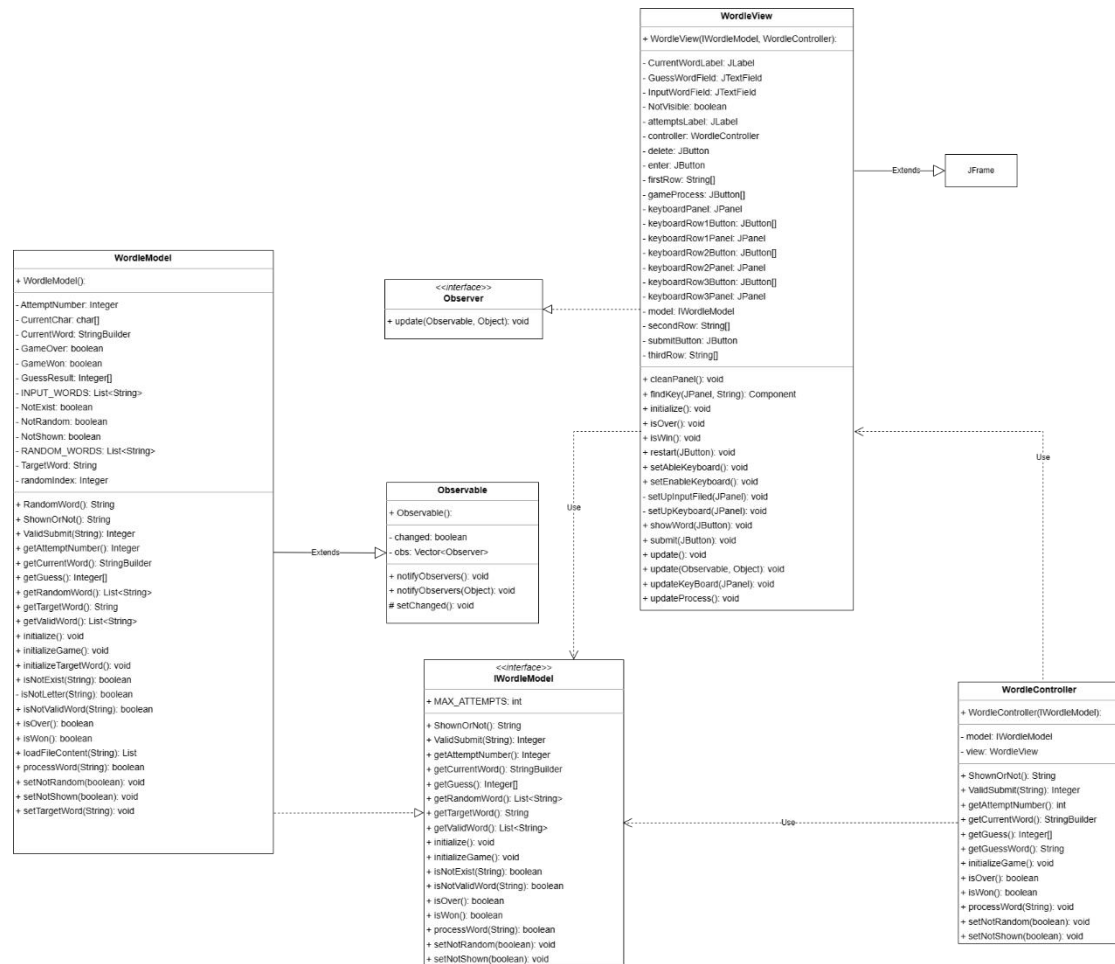


Figure 1 Class Diagram

2. Implement

3. Testing

a) GUI Testing

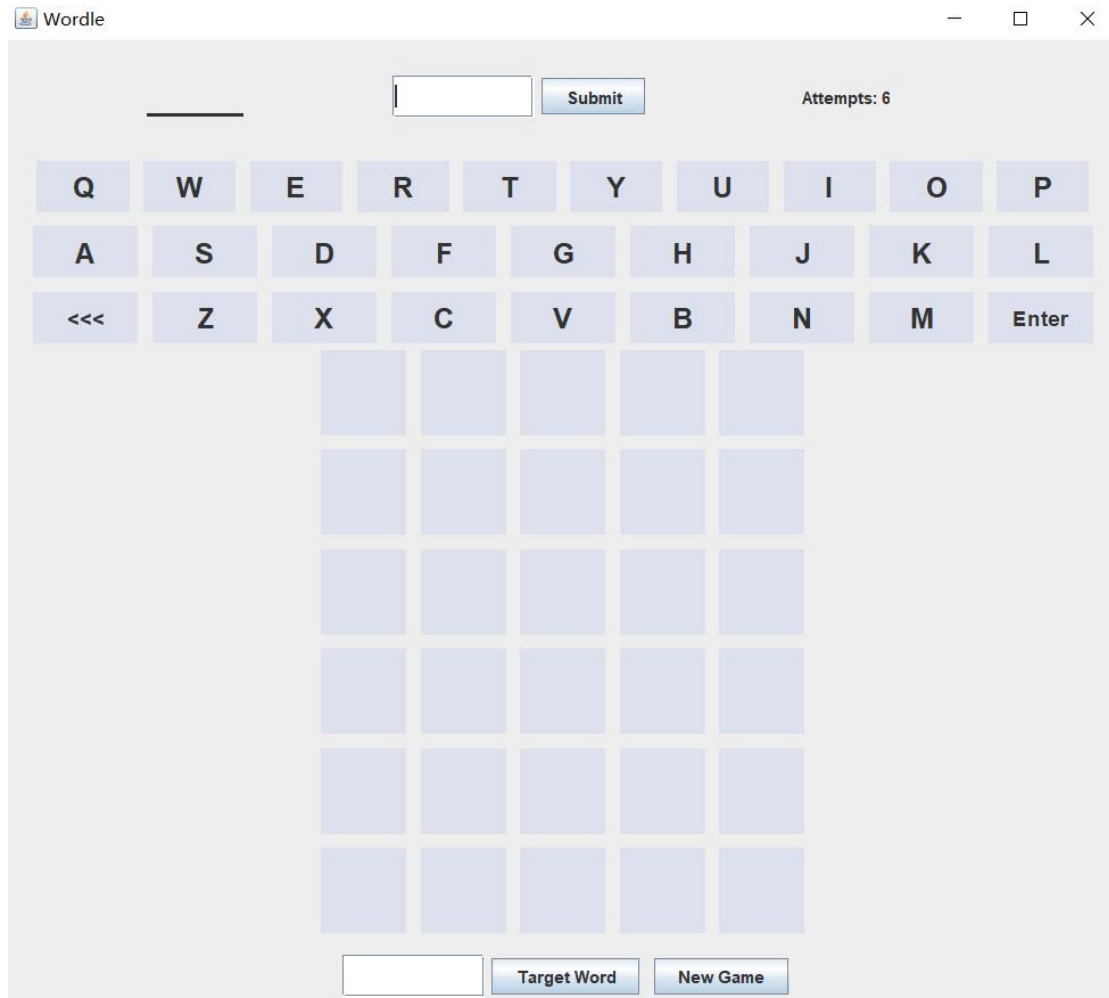


Figure 2 The initialization of GUI Wordle

Figure 2 shows a GUI version that complies with **FR4**. Backspace, enter, and all letters are on the keyboard. When the player clicks a letter on the keyboard, the letter is displayed back in the word input field, and the player can click on "<<<" to remove the last typed letter on a non-empty line. In addition, the word typed can be submitted by clicking "Enter".

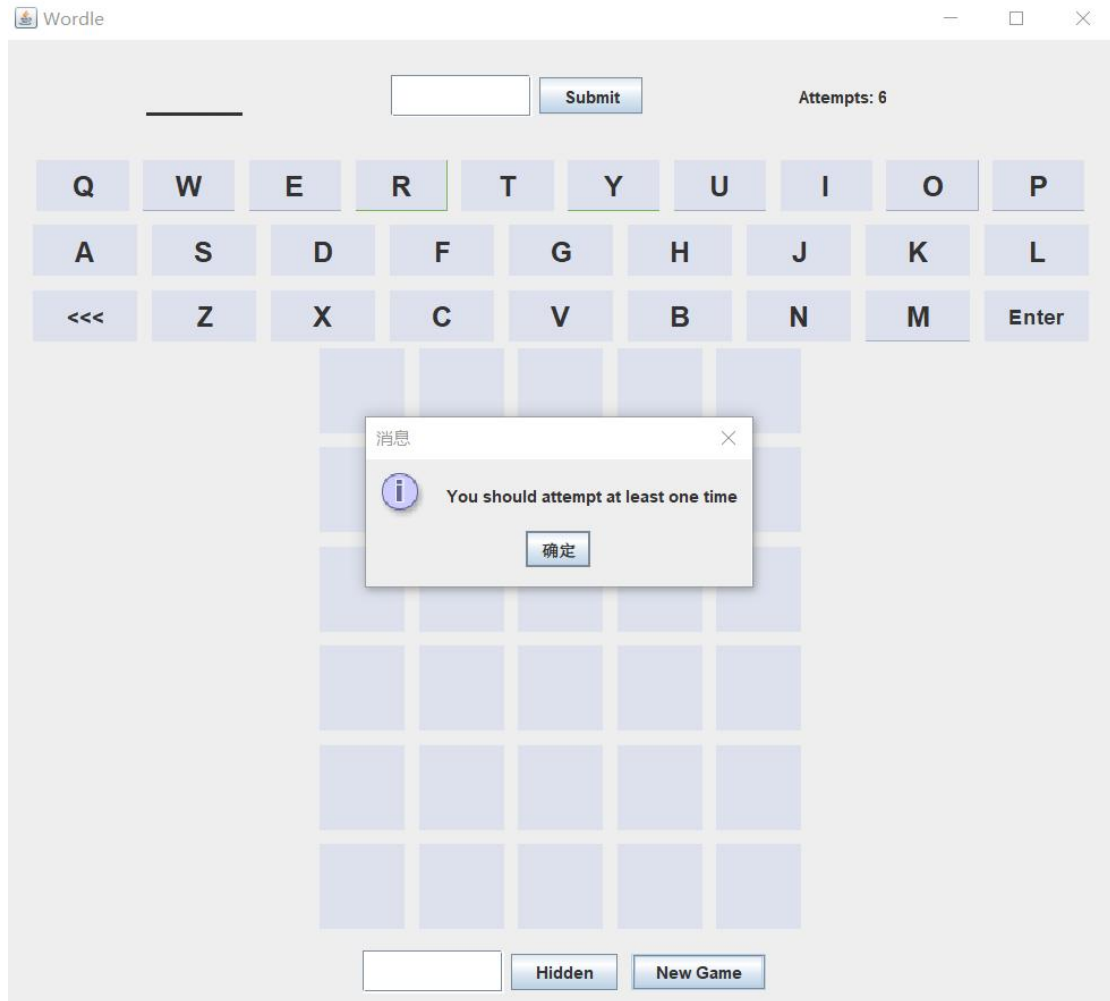


Figure 3 Restart the game after no valid word guesses

Figure 3 shows a GUI version that complies with **FR7**. GUI version provides the button to start a new game which is enabled only after the first valid guess has been made.

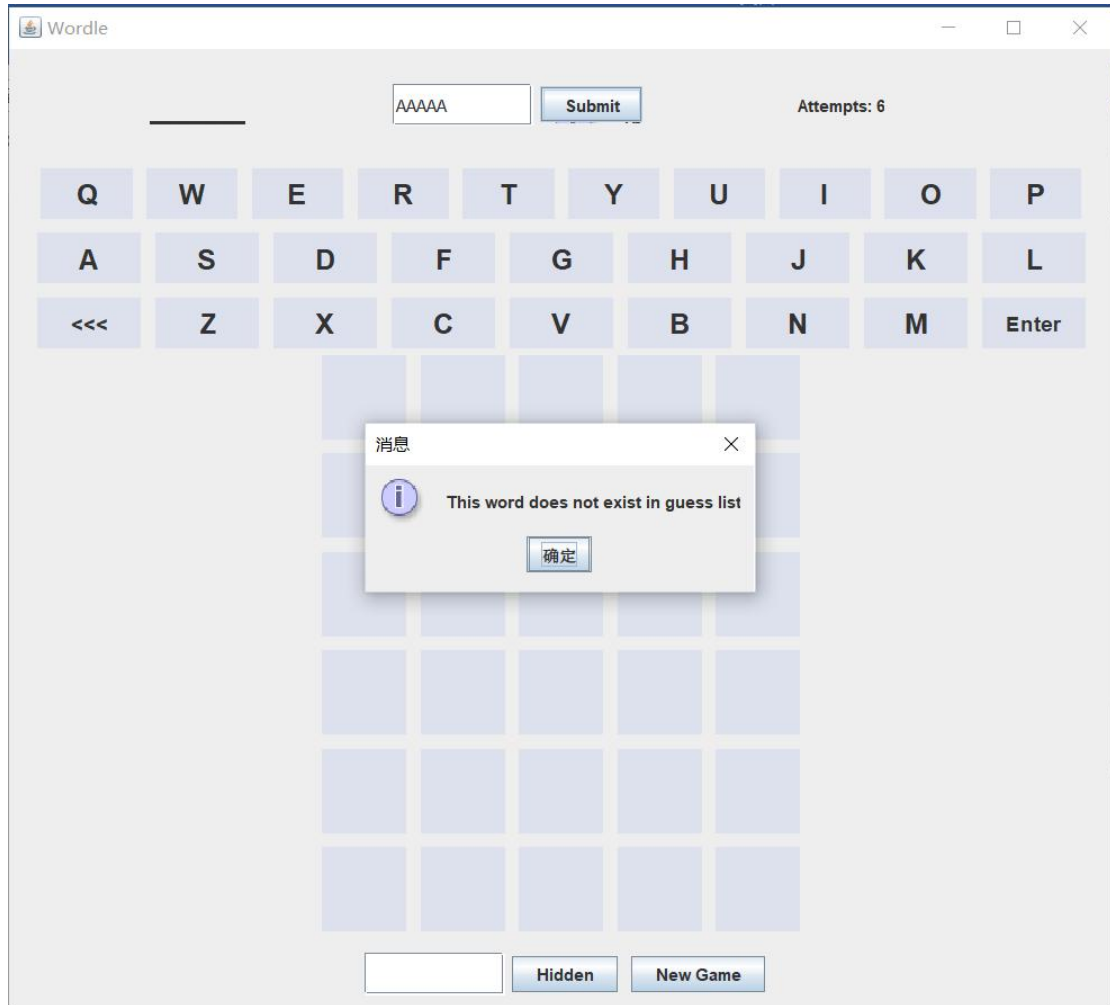


Figure 4 the message prompting user to guess the word included in guess list

Figure 4 shows a GUI version that complies with **FR3-flag1**.

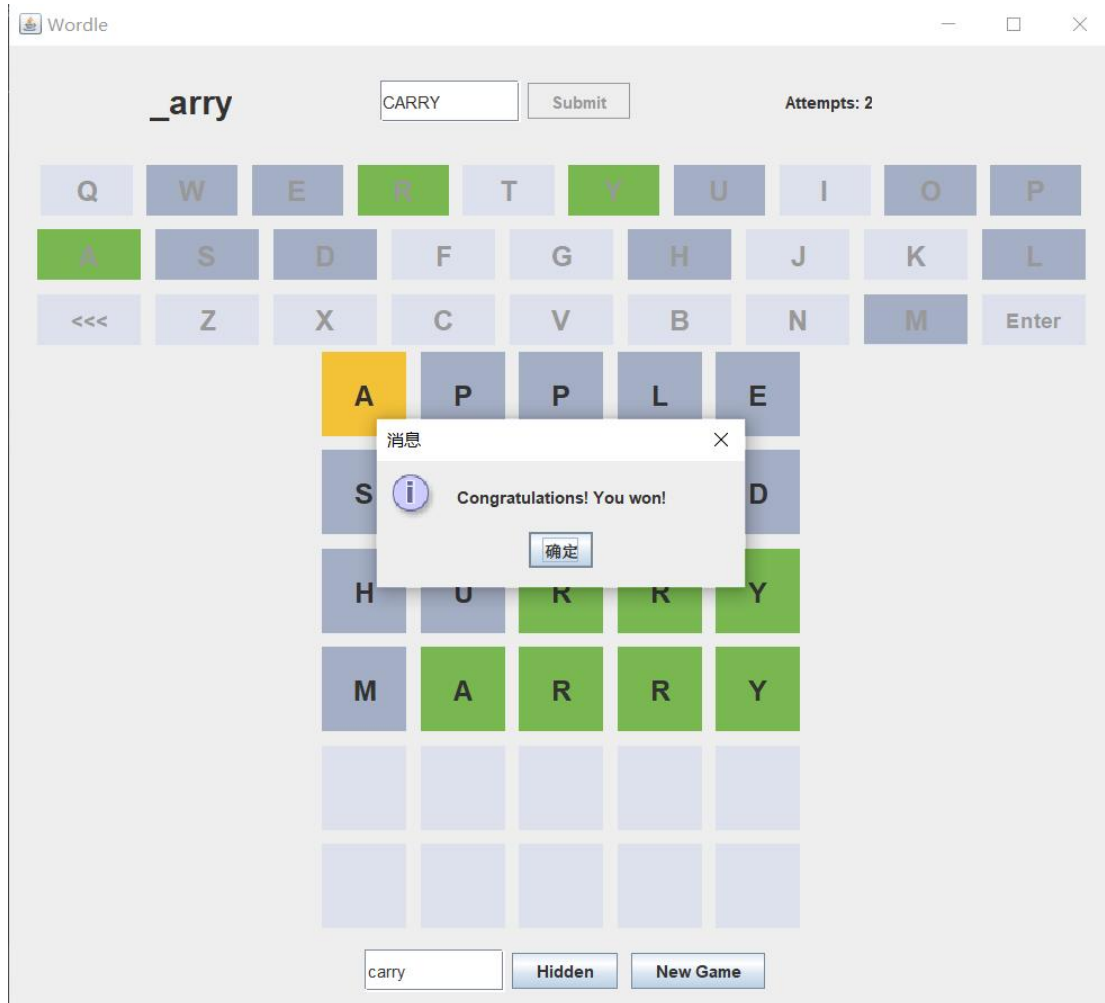


Figure 5 User guesses the word correctly on the fourth time

Figure 5 shows a GUI version that complies with **FR6 and FR3-flag2**. the keyboard's color can vary according to the accuracy of the target word and the guess match. Each guess and the matching outcome will be shown on the panel. In addition, the player can click "Hidden" to hide the target word and click "Show" to see the target word.

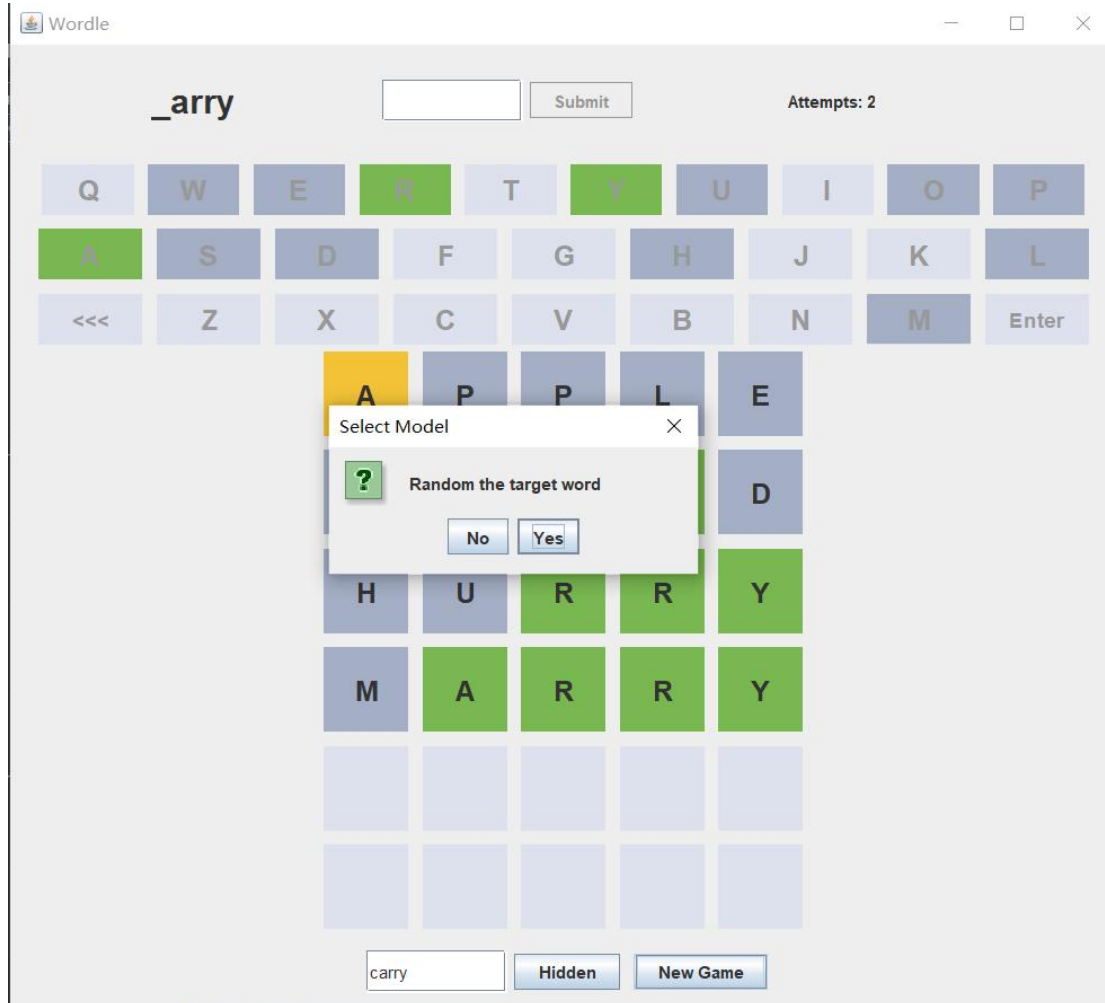
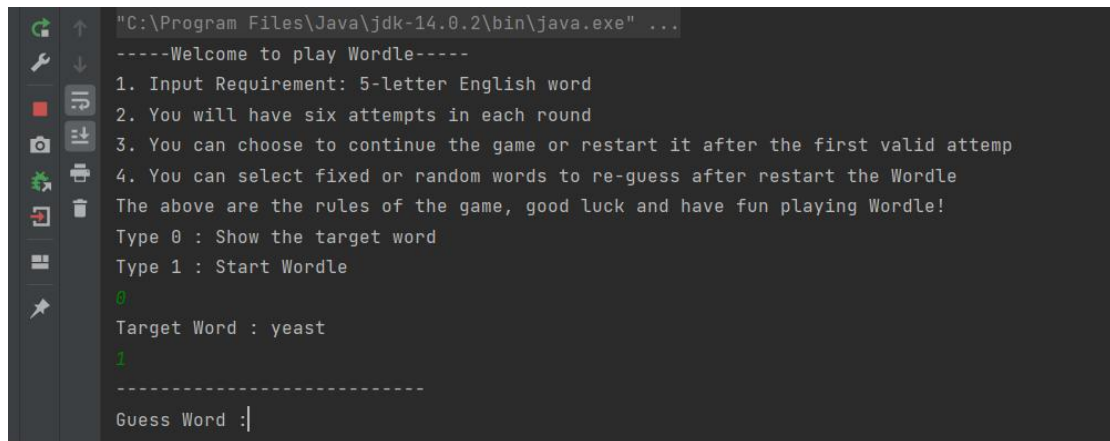


Figure 6 A choice of fixed target words or random target words appears after click “New Game”

Figure 6 shows a GUI version that complies with **FR7** and **FR3-flag2**. After a valid attempt, the player can restart the game and choose to continue guessing the word or to re-random the target word. After choosing to re-random the target word, the target word will be re-generated and visible, otherwise the target word will be the same as the target word from the previous round.

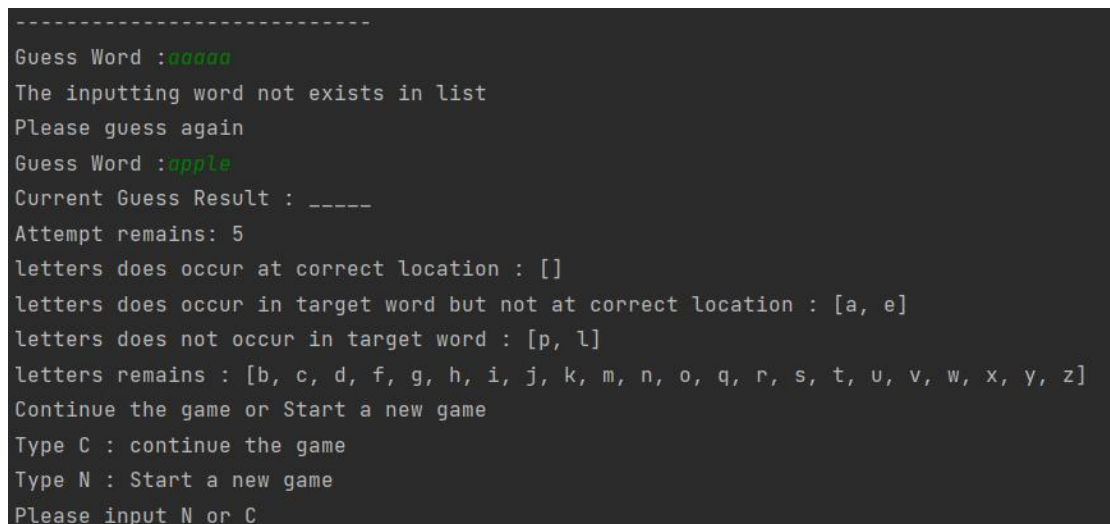
b) CLI Testing



```
"C:\Program Files\Java\jdk-14.0.2\bin\java.exe" ...
-----Welcome to play Wordle-----
1. Input Requirement: 5-letter English word
2. You will have six attempts in each round
3. You can choose to continue the game or restart it after the first valid attempt
4. You can select fixed or random words to re-guess after restart the Wordle
The above are the rules of the game, good luck and have fun playing Wordle!
Type 0 : Show the target word
Type 1 : Start Wordle
0
Target Word : yeast
1
-----
Guess Word :|
```

Figure 7 CLI version of initializing the Wordle

Figure 7 shows the CLI version that complies with **FR3-flag2**. The player can type 0 to check the target word and type 1 to start the Wordle.



```
-----
Guess Word :aaaaa
The inputting word not exists in list
Please guess again
Guess Word :apple
Current Guess Result : -----
Attempt remains: 5
letters does occur at correct location : []
letters does occur in target word but not at correct location : [a, e]
letters does not occur in target word : [p, l]
letters remains : [b, c, d, f, g, h, i, j, k, m, n, o, q, r, s, t, u, v, w, x, y, z]
Continue the game or Start a new game
Type C : continue the game
Type N : Start a new game
Please input N or C
```

Figure 8 The guess process while the player input “aaaaa” and “apple”

Figure 8 shows the CLI version that complies with **FR3-flag1** and **FR6**. The attempt cannot be counted as one of the tries while the player input the word which is not included in the list of known words. The console indicates available letters by listing the letters in four separate categories which are correct letters, include but incorrect letters, error letters and remain letters. In addition, after the first try, the player has choices of continuing the game or start a new game.


```

Continue the game or Start a new game
Type C : continue the game
Type N : Start a new game
Please input N or C
N
Please Select model: Type F: word is fixed; Type R: word is random
F
-----Welcome to play Wordle-----
1. Input Requirement: 5-letter English word
2. You will have six attempts in each round
3. You can choose to continue the game or restart it after the first valid attempt
4. You can select fixed or random words to re-guess after restart the Wordle
The above are the rules of the game, good luck and have fun playing Wordle!
Type 0 : Show the target word
Type 1 : Start Wordle
0
Target Word : yeast

```

Figure 9 World provides the choice to start a new game

Figure 9 shows the CLI version that complies with **FR3-flag3**. The player can select the choice to fix or random the target word.

```

-----
Guess Word :earth
Current Guess Result : s_o__
Attempt remains: 0
letters does occur at correct location : [s, o]
letters does occur in target word but not at correct location : [w, h]
letters does not occur in target word : [a, p, l, e, r, d, y, t]
letters remains : [b, c, f, g, i, j, k, m, n, q, u, v, x, z]
Sorry! You lost! The word is shown

Process finished with exit code 0

```

Figure 10 Game over while the player had guessed unsuccessfully

Figure 10 shows the CLI version that complies with **FR2**. When the player has not guessed correctly after six guesses, the console will show that the game has failed, print the target word and end the program.

```

"C:\Program Files\Java\jdk-14.0.2\bin\java.exe" ...
----Welcome to play Wordle----
1. Input Requirement: 5-letter English word
2. You will have six attempts in each round
3. You can choose to continue the game or restart it after the first valid attempt
4. You can select fixed or random words to re-guess after restart the Wordle
The above are the rules of the game, good luck and have fun playing Wordle!
Type 0 : Show the target word
Type 1 : Start Wordle
0
Target Word : prior
1
-----
Guess Word :prior
Congratulations! You won!

Process finished with exit code 0

```

Figure 11 Game over while the player had guessed successfully

Figure 11 shows the CLI version that complies with **FR2**. When the player has guessed correctly, the console will show that the game has won and end the program.

c) Junit Testing

```

//WordleModelTest.java
assertEquals( expected: "c__at", model.getCurrentWord().toString());
//Assert: Verify if game is over
assertFalse(model.isOver());

//Act: Perform the processing word and set up the guess word as "eclat" for the
model.processWord( GuessWord: "eclat");
//Assert: Verify if guess result equals "[1, 1, 1, 2, 2]"
assertEquals( expected: "[1, 1, 1, 2, 2]", Arrays.toString(model.getGuess()));
//Assert: Verify if attempt number remains 3
assertEquals( expected: 3, model.getAttemptNumber());
//Assert: Verify if game is over
assertFalse(model.isOver());

//Act: Perform the processing word and set up the guess word as "cleat" for the
model.processWord( GuessWord: "cleat");
//Assert: Verify if attempt number remains 2
assertEquals( expected: 2, model.getAttemptNumber());

```

Test Scenario	Duration	Status
WordleModelTest (game)	66 ms	Passed
TestScenario1()	42 ms	Passed
TestScenario2()	10 ms	Passed
TestScenario3()	14 ms	Passed

Tests passed: 3 of 3 tests - 66 ms

Process finished with exit code 0

Figure 12 Junit tests in three different test scenarios