1 Overview

A word game named WORDLE that has gained its popularity in the recent times.

The game WORDLE is challenging, but simple: Once a day, players have six guesses to identify a new five-letter word (all players receive the same word on a given day). Each guess provides color-coded hints: a letter turns green if it is in the correct spot, yellow if it is part of the word but in a different spot, and gray if it is not in the word at all. An example to a single play is illustrated in Figure 1.

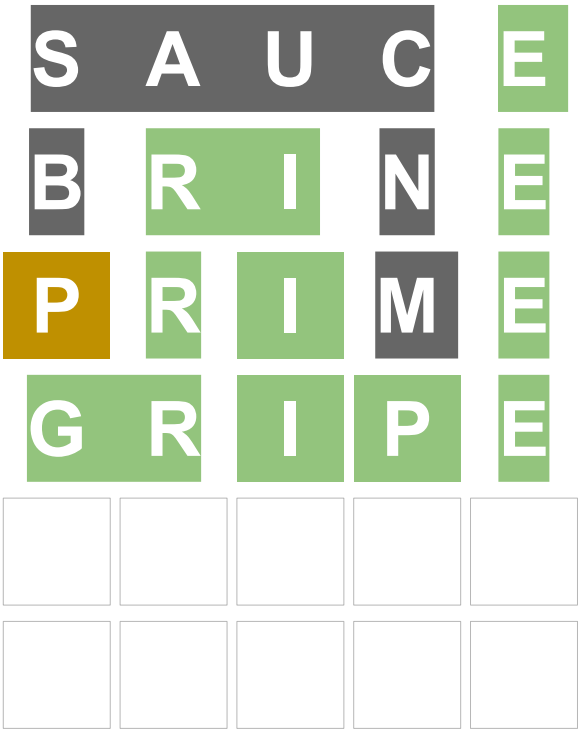

Figure 1: An illustration on how to play WORDLE game.

In this assignment, you are expected to develop a simplified version of this game. To do that follow the instructions below.

Instructions

- Create a class `Wordle`.
- Load the word set from `dict.txt` file[2] to a **String array**. Note that the file consists of **2317 words**, so you can adopt **hard-coding** fashion to set the size of array fixed.
- Among these words, **randomly** select one word as **key word**.

- Enable user to provide **six words** as command-line argument to your program.
- For each word, your program must first check if the length of the provided word is less or greater than **five**. If so, it should print out "**The length of word must be five!**".
- What user provides as a word must be in dictionary (i.e., in the array) as the true case in WORDLE game. So, your program must check if it exists. If it does not exist, it should print out "**Word does not exist in the dictionary!**".
- Once **key word** is guessed right by the user before her/his last shot, the following message should be displayed and no more input is evaluated:

  ```
  Congratulations! You guess right in kth shot!
  ```

  Note here that k is a counter for try (i.e., $1 \leq k \leq 6$), and so the suffixes to represent the ordinal indicator (e.g., *-st, -nd, -rd, or -th*) must be consistent with the number k. (e.g., "**Congratulations! You guess right in [1st, 2nd, 3rd, 4th, ...] shot!**")
- If none of the above criteria is satisfied, it is time for your program to examine the word letter by letter. For key word **`BASIC`** and user input **`DASNB`**[3], the output should be as follows:

  ```
  1. letter does not exist.
  2. letter exists and located in right position.
  3. letter exists and located in right position.
  4. letter does not exist.
  5. letter exists but located in wrong position.
  ```

- When user failed to find the **key word** after he/she run out of all tries, your program should print out "**You failed! The key word is `KEY`.**"; where **`KEY`** is a placeholder that contains the **key word**.

Command for Running the Application

```
java Wordle WORD1 WORD2 WORD3 WORD4 WORD5 WORD6
```

Demonstrations

Example 1

When your program is run with **`java Wordle LONDON PARIS SAUCE BRINE PRIME GRIPE`** command, it should produce the following output. Note that the output of your program has to match with the one given here including the printing format shown below.

```
Try1 (LONDON): The length of word must be five!
Try2 (PARIS): The word does not exist in the dictionary!
```

[3] assume it exists in the dictionary.

```
Try3 (SAUCE):
1. letter does not exist.
2. letter does not exist.
3. letter does not exist.
4. letter does not exist.
5. letter exists and located in right position.
Try4 (BRINE):
1. letter does not exist.
2. letter exists and located in right position.
3. letter exists and located in right position.
4. letter does not exist.
5. letter exists and located in right position.
Try5 (PRIME):
1. letter exists but located in wrong position.
2. letter exists and located in right position.
3. letter exists and located in right position.
4. letter does not exist.
5. letter exists and located in right position.
Try6 (GRIPE): Congratulations! You guess right in 6th shot!
```

Example 2

When your program is run with **`java Wordle NEVER PAUSE SOUCE SAUCE BRINE PRIME GRIPE`** command, it should produce the following output. Note that the output of your program has to match with the one given here including the printing format shown below.

```
Try1 (NEVER):
1. letter does not exist.
2. letter exists but located in wrong position.
3. letter does not exist.
4. letter exists but located in wrong position.
5. letter does not exist.
Try2 (PAUSE):
1. letter does not exist.
2. letter exists but located in wrong position.
3. letter does not exist.
4. letter does not exist.
5. letter exists but located in wrong position.
Try3 (SOUCE): The word does not exist in the dictionary!
Try4 (SAUCE):
1. letter does not exist.
2. letter exists but located in wrong position.
3. letter does not exist.
4. letter does not exist.
5. letter exists but located in wrong position.
Try5 (BRINE):
1. letter does not exist.
```

```
2. letter does not exist.
3. letter exists but located in wrong position.
4. letter does not exist.
5. letter exists but located in wrong position.
Try6 (PRIME):
1. letter does not exist.
2. letter does not exist.
3. letter exists but located in wrong position.
4. letter does not exist.
5. letter exists but located in wrong position.
You exceeded maximum try shot!
You failed! The key word is IDEAL.
```