1 Wordle

For this project, you will be recreating the popular game *Wordle*! For those who haven't played, *Wordle* is a word game in which the player has six attempts to guess a secret 5-letter word. After each guess, the user is given information regarding how each letter of their guess fits into the secret word (i.e. whether or not each letter is in the word and/or in the correct spot).

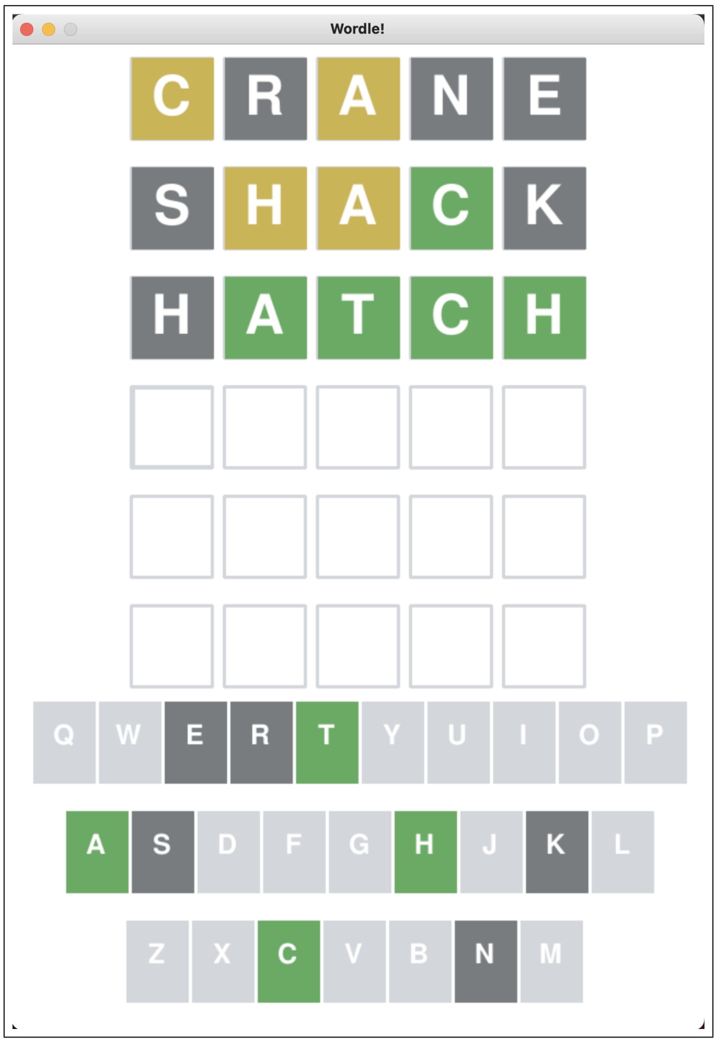

Figure 1: Example for the secret word *"LATCH"*.

Of the code provided, the only file that you are allowed to modify is *wordle_starter.py*, which contains starter code for the project. Everything related to the graphical user interface (GUI) will be handled for you. To start the program and boot up the GUI, simply run *wordle_starter.py*. Your job will be to program the logic behind Wordle, which requires your implementation of three methods, outlined in the following subsections.

1.1 Part 1: Initialize the Game (10 pts.)

To select a secret word and to determine whether or not a guess is valid, *Wordle* uses a long list of "valid words." The file *words.txt* (provided for you) contains over two thousand valid 5-letter words, one per line. Your job is to implement the *init_game()* method, which accepts the file path (of type *string*) as input, and outputs a list of all the words in the file, along with a randomly selected secret word. Implement the following inside of the *init_game()* method:

1. Randomly select a word from the list.
2. Return the list and the secret word.

1.2 Part 2: Verify a Guess (10 pts)

Next, it is necessary to check whether or not a user's guess is valid. For this part, your job is to implement the *checkValidInput()* method, which accepts the user's guess and the list of valid words as input, and outputs *True* if the guess is valid, and *False* otherwise. Implement the following inside of the *checkValidInput()* method:

1. Check that the player's guess is exactly 5 letters long.
2. Check that the player's guess is in the list of valid words.
3. If both of the above conditions are satisfied, return *True.* Otherwise, return *False.*

1.3 Part 3: Evaluate a Guess (20 pts)

Now that you have verified that a guess is valid, we need to evaluate the guess and tell the user how they did. In *Wordle*, each letter in the guess is evaluated to be: *a)* correct letter in the correct position, *b)* correct letter in the wrong position, or *c)* incorrect letter. Your job is to implement the *eval_guess()* method, which accepts the secret word and the user's guess as input, and outputs a letter-by-letter evaluation of the guess. Implement the following inside of the *eval_guess()* method:

1. Loop through each letter of the guess and determine how it should be evaluated, based on the secret word.
2. Construct an output string that contains a '$' for each correct letter in the correct position, a '*' for each correct letter in the incorrect position, and a '#' for each incorrect letter. For example, if the secret word is *"CRANE"* and the user guesses *"LEARN"*, the evaluation should be the string *"#*$**"*.
3. Return the evaluation string, as constructed in the previous step.

1.4 Code Submission

Congratulations – you just recreated the game *Wordle*! The only code that you must include in your submission is the *wordle_starter.py* file.

1.5 Questions (10 pts)

1.5.1 For Part 1, how did you ensure that you selected a random word? (2 pts)

1.5.2 For Part 2, how did you ensure both conditions were true? (3 pts)

1.5.3 For Part 3, what challenges did you face when handling this portion of the project? How did you check that you were correctly evaluating the guess? (5 pts)