Intro to Computer Science, Wahluke High School

Unit 3 Project, Option 2: Wordle Solver

This assignment is an opportunity to apply your new skills in creating custom functions to implement a solver for Wordle, a trending word-guessing game! This is an advanced project that ambitious students should work on.

1. Overview

Wordle is a game where a five-character word is randomly selected, and the player must guess the target word correctly. Every time the player makes a guess, the color of the character tiles change to provide a hint: green tiles indicate the character is in the correct position, yellow tiles indicate the character is in the target word but not in that specific position, and grey tiles indicate the character is not in the target word at all.

You can play it yourself if you're not familiar with Wordle:

- English version: https://www.powerlanguage.co.uk/wordle/
- Spanish version: https://wordle.danielfrg.com/

In this project, you will implement a Wordle solver in Python using functions. This prompt specifies five helper functions that are required for the Wordle solver. Each function is small and well-defined. When all the helper functions are done, the hard part is done, and the five functions can be used together as building blocks to a completed Wordle solver!

(continued...)

2. Example Program Output

```
Welcome to the Wordle solver!
First, you will enter characters that are known to be at different
positions in your word.
Second, you will enter characters that are known to not be in your
word.
Finally, the program will list out all the words that match your
criteria!

What character is at position 0? (empty for unknown) f
What character is at position 1? (empty for unknown) r
What character is at position 2? (empty for unknown)
What character is at position 3? (empty for unknown)
What character is at position 4? (empty for unknown) s
What characters are not in your word? e
['frabs', 'frags', 'fraps', 'frass', 'frats', 'fraus', 'frays',
'fribs', 'frigs', 'frits', 'frogs', 'frons', 'frows', 'frugs']
Want to play again? (y/n) y

What character is at position 0? (empty for unknown) f
What character is at position 1? (empty for unknown) r
What character is at position 2? (empty for unknown)
What character is at position 3? (empty for unknown)
What character is at position 4? (empty for unknown) s
What characters are not in your word? eaog
['fribs', 'frits']
Want to play again? (y/n) y

What character is at position 0? (empty for unknown) s
What character is at position 1? (empty for unknown) h
What character is at position 2? (empty for unknown) o
What character is at position 3? (empty for unknown)
What character is at position 4? (empty for unknown) k
What characters are not in your word?
['shock', 'shook']
Want to play again? (y/n) n
```

(continued...)

3. Matching Format

When playing Wordle, the characters you selected turn green if they are the correct character in the correct position.

These allow us to specify a *format*, a five-element list defining characters that must be in specific positions in the target word. If we don't know what character is at a specific position, any character can be at that position. Here are some examples:

Format	Matching Words
`['s', 'h', 'o', '', 'k']`	`['shock', 'shook']`
`['f', 'r', '', '', 's']`	`['frabs', 'frags', 'fraps', 'frass', 'frats', 'fraus', 'frays', 'frees', 'frets', 'fribs', 'fries', 'frigs', 'frits', 'froes', 'frogs', 'frons', 'frows', 'frugs']`
`['g', 'o', 'o', '', '']`	`['goose', 'goody', 'gooey', 'goofy', 'gooby', 'goods', 'goofs', 'googs', 'gooks', 'gooky', 'goold', 'gools', 'gooly', 'goons', 'goony', 'goops', 'goopy', 'goors', 'goory', 'goosy']`

We will create two helper functions:

- `does_word_match_format`
  - This function should accept two parameters:
    - `word` – a specific word to evaluate
    - `fmt` – a five-element list defining characters that must be in specific positions in the word
  - This function should use a loop to look at every character in word and fmt to evaluate whether the provided word matches the provided format.
  - This function should return a boolean: True if the provided word matches the provided format, False otherwise.

- `filter_to_words_matching_format`
  - This function should accept two parameters:
    - `wordlist` – a list of all possible words
    - `fmt` – a five-element list defining characters that must be in specific positions in the filtered words
  - This function should use a loop to (1) look at every word in wordlist, (2) call does_word_match_format for each word, and (3) generate a list of results containing all words matching the format.
  - The function should return a list of words from wordlist which match the provided format.

4. Excluded Characters

When playing Wordle, the characters you selected turn grey if the character is not anywhere in the target word.

These allow us to specify a list of excluded characters which cannot be in the target word. Here are some examples:

Format	Excluded Characters	Matching Words
`['f', 'r', '', '', 's']`	`[]`	`['frabs', 'frags', 'fraps', 'frass', 'frats', 'fraus', 'frays', 'frees', 'frets', 'fribs', 'fries', 'frigs', 'frits', 'froes', 'frogs', 'frons', 'frows', 'frugs']`
`['f', 'r', '', '', 's']`	`['e', 'a']`	`['fribs', 'frigs', 'frits', 'frogs', 'frons', 'frows', 'frugs']`
`['f', 'r', '', '', 's']`	`['e', 'a', 'o', 'g']`	`['fribs', 'frits']`

(continued...)

We will create three helper functions:

- `does_word_include_char`
    - This function should accept two parameters:
        - `word` – a specific word to evaluate
        - `char` – a character to check for in the provided word
    - This function should use a loop to look at every character in word and evaluate whether the provided word contains the character.
    - This function should return a boolean: True if the provided word includes the provided character, False otherwise.

- `filter_to_words_excluding_char`
    - This function should accept two parameters:
        - `wordlist` – a list of all possible words
        - `excluded_char` – a character that must be excluded in the target word
    - This function should use a loop to (1) look at every word in wordlist, (2) call does_word_include_char for each word, and (3) generate a list of results containing all words which do not include the excluded character.
    - The function should return a list of words from wordlist which do not include the excluded character.

- `filter_to_words_excluding_all_chars`
    - This function should accept two parameters:
        - `wordlist` – a list of all possible words
        - `all_excluded_chars` – a list of character that must be excluded in the target word
    - This function should use a loop to (1) look at every word in all_excluded_chars, (2) call filter_to_words_excluding_char for each excluded character, and (3) generate a list of results containing all words which do not include any of the excluded characters.
    - The function should return a list of words from wordlist which do not include any of the excluded characters.

(continued...)

5. Import Word Provider

The instruction team has prepared a word_provider library which you can use to get the full wordlist. This is a great example of how functions and libraries are helpful: you can use custom functions created by your instructors just by importing them!

At the top of your Python file, add one of these lines of code to import either of the get_all_words functions:

```
from word_provider import get_all_words_en
from word_provider import get_all_words_es
```

Then, you'll be able to call the function like this:

```
all_words = get_all_words_en()
all_words = get_all_words_es()
```

You may implement your game using either the English or Spanish wordset.

These are the function contracts for the two provided functions:

```
# Name: get_all_words_en
# Purpose: Get all 5-character English words
# Inputs: none
# Output: list of all 5-character English words

# Name: get_all_words_es
# Purpose: Get all 5-character Spanish words
# Inputs: none
# Output: list of all 5-character Spanish words
```

(continued...)

6. User Interface

Good job – you've written all the logic! Now, write the code required to (1) get user input on the format and excluded characters, (2) use that user input to call your custom functions, and (3) display the matching words to the user.

You can look at the sample program output for an example of what the user interface may behave like. You can implement any kind of user interface you'd like, but it must meet these requirements:

- Provide reasonably clear instructions
- Be reasonably easy to use
- Prompt the user to provide the known characters to specify the format
- Prompt the user to provide the excluded characters
- Show a list of matching characters after receiving the inputs
- Allow the user to re-run the program if they'd like to

7. Style

For each of your functions, please include a comment in this format to make your code easier to read:

```
# Name:
# Purpose:
# Inputs:
# Outputs:
```

8. Resources

In addition to the lab assignments you've completed and class notes you've taken, here are some resources for relevant course topics

- 2.4 & 2.5 – lists cheat sheet: https://bit.ly/24-reference
- 2.7 – while loops: https://tealsk12.github.io/2nd-semester-introduction-to-computer-science/readings.md.html#associatedreadings/2.7
- 3.1 – 3.4 – readings: https://tealsk12.github.io/2nd-semester-introduction-to-computer-science/readings.md.html#associatedreadings/unit3-functions

9. Challenge

These challenges are opportunities to independently develop your computer science skills by extending your program with interesting, advanced functionality. Note that these challenges are optional and are not eligible for a grade. Please complete the rest of the assignment thoroughly before working on the challenges.

- Wordle also has yellow tiles, which are characters which are in the target word, but were guessed to be in an incorrect position. Incorporating this into your project would make your Wordle solver more effective by filtering to words which include specific characters somewhere in the word. How could you incorporate yellow tiles into your project?

- The English and Spanish wordsets are both provided. How could you let the user choose which language to use?

(continued...)

10. Rubric

Please submit this assignment on Repl. If you have questions or issues, please don't hesitate to ask any member of the teaching team, or email `help@teals.dev`

The rubric is a helpful tool to make sure you've completed all parts of the assignment.

Rubric Item	**Points**
3. Matching Format	10
4. Excluded Characters	15
5. Import Word Provider	1
6. User Interface	9
7. Styles	5
Assignment Total	40