W04 Project: Word Puzzle

Regarding Milestones:

A milestone or milepost is a marker placed along a highway to tell you how far you have come, or to indicate your progress toward your destination. In software development projects, a Project Milestone marks a specific point along a project timeline.

To help you make progress toward finishing this project, you will complete part of the program during the middle of the week and submit a "Milestone Submission." Then, by the end of the week, you will complete the program and submit the finished version.

You should read over the complete project description first. Then, at the bottom of this page, you will see which features are required for the milestone and which are required for the overall project.

Overview

Many games and puzzles require iteration where a person (or a computer) repeats the same steps for each piece in the game or each spot in a puzzle. For this assignment, you will create an interactive word puzzle game that allows the user to make guesses until they get the answer correct, and hints are provided along the way.

Wordle is a web-based word game that became popular online in the early part of 2022. (You can learn more about it from the Wordle Wikipedia page. If you want, you can play it online for free from the New York Times page.)

For this assignment, you'll create a puzzle game that follows a similar pattern.

Project Description

The program contains a hidden secret word stored in a variable. This word can have any number of letters in it. When the program runs, the user is shown underscores (**_**) for each letter of the word.

The user then enters a guess. If the guess is correct, the user wins and the game is over.

If the guess is not correct, the user will be given a hint to help them improve their guess for the next round.

The game continues prompting the user for new guesses and showing them hints until they guess the word correctly. When the game is over, the program displays the number of guesses that were made.

The guess must be the same number of letters as the secret word. If the guess has a different numbers of letters, the user is given a message explaining this, and no hint is provided.

The hint shows the letters of the guess, but each letter is rendered in a special way as follows:

- An underscore **_** indicates that the letter was not present in the secret word.
- A lowercase letter indicates that the letter was present somewhere in the secret word, but not at that position.
- An uppercase letter indicates that the letter was present at that exact spot in the secret word. (In other words, if the second letter in the guess is also the second letter in the secret word, then that letter is shown as a capital.)

For example, if the secret word were: **`mosiah`**, then the program would initially display:

```
Your hint is: _ _ _ _ _ _
```

If the user's guess were: **`temple`**, they would learn that the **`m`** in **`temple`** is the only letter in the secret word, but it is in the wrong spot, so it would be displayed in lowercase as shown:

```
Your hint is: _ _ m _ _ _
```

But if the user instead guessed: **`moroni`**, they would receive the following as a hint:

```
Your hint is: M O _ o _ i
```

Notice that in the hint above, the **`M`** and the first **`o`** from **`moroni`** are capitalized, because those letters are also the first two letters of **`mosiah`**.

The **`i`** and the second **`o`** from moroni are displayed, because they are present in the secret word, but they are shown in lowercase, because the the letters in the secret word at that location are different.

The **`r`** and the **`n`** from **`moroni`** are not displayed, but instead an underscore is shown in each place, because these letters are not present in the secret word in any location.

A sample of the complete program might look as follows:

```
Welcome to the word guessing game!

Your hint is: _ _ _ _ _ _
What is your guess? temple
Your hint is: _ _ m _ _ _
What is your guess? moroni
Your hint is: M O _ o _ i
What is your guess? hhhhhh
Your hint is: h h h h h H
What is your guess? mosiah
Congratulations! You guessed it!
It took you 4 guesses.
```

Because capital letters have meaning in our hints, the secret word should be all lower case. Similarly, when the user enters their guess, you should convert it to lowercase prior to checking for matches.

If the guess has a different number of letters than the secret word, it still counts as a guess, but the user does not receive a hint. This is shown in the following example:

```
Welcome to the word guessing game!

Your hint is: _ _ _ _ _ _
What is your guess? nephi
Sorry, the guess must have the same number of letters as the secret wor

What is your guess? a
Sorry, the guess must have the same number of letters as the secret wor
```

```
What is your guess? helaman
Sorry, the guess must have the same number of letters as the secret wor

What is your guess? abcdefghijklmnopqrstuvwxyz
Sorry, the guess must have the same number of letters as the secret wor

What is your guess? temple
Your hint is: _ _ m _ _ _
What is your guess? mosiah
Congratulations! You guessed it!
It took you 6 guesses.
```

Video Demo

The following video walks through a demonstration of the program so that you can understand the rules of the game and how it is supposed to work:

- Wordle Program Demo (6 minutes)

Milestone Requirements

By the middle of the week, to help make sure you are on track to finish the assignment, you need to complete the following:

1. Have a secret word stored in the program.
2. Prompt the user for a guess.
3. Continue looping as long as that guess is not correct.
4. Calculate the number of guesses and display it at the end.

- You do not need to have any hints at this point.

The following shows the functionality of the program at this point:

```
Welcome to the word guessing game!

What is your guess? temple
Your guess was not correct.
What is your guess? moroni
Your guess was not correct.
What is your guess? mosiah
Congratulations! You guessed it!
It took you 3 guesses.
```

Final Requirements

In addition to the Milestone requirements, your final program must:

1. Use a loop to generate the initial hint.
2. Add a check to verify that the length of the guess is the same as the length of the secret word. If it is not, display a message. If they are the same, then proceed to give the hint.
3. Add the hints according to the rules above.
4. Make sure to account for the following in your hints:
    - Letters that are not present at all in the secret word (underscore _).
    - Letters that are present in the secret word, but in a different spot (lowercase).
    - Letters that are present in the secret word at that exact spot spot (uppercase).
5. Additionally, be sure to add a space after every character in the hint as in the examples above.

Showing Creativity and Exceeding Requirements

Once you have the basic rules of the game in place, consider adding something extra to the hints, other rules or limitations to the number of guesses, or anything else you

think would be fun.

If you want to look ahead at lists or files, you could start the game with a list of words, and select a random one for the secret word.

> **Important:** In order to receive credit for showing creativity, you must include a comment at the top of the program that describes in 1-2 sentences what you have added.

Submission

There is an assignment submission in Canvas for both the Milestone and the Final Submission. Make sure to:

- Return to Canvas to submit:
    1. The milestone submission (midweek)
    2. The finished project (end of week)

Up Next

- W04 Reflection

Other Links:

- Return to: Week Overview | Course Home