While Loops Challenge 29: Guess My Word App

Description:
You are responsible for writing a program that plays a word guessing game with a user. Your program will provide a category of words to the user and a string of dashes “-----” that represent the length of the word. The user will guess the word and with each incorrect guess, your program will reveal a letter at random, “-a---”. Upon guessing the word correctly, your program will then inform the user how many guesses they took.

Step By Step Guide:

- Print a welcome message.
- Create a dictionary called game_dict that has a minimum of four keys.
    - Each key should be a category such as "sports", "colors", or "fruits".
    - Each subsequent value should be a list.
    - In the list you should have a minimum of 6 strings that are in that category.
    - For instance if the key is sports, the subsequent value should be a list containing the words "basketball", "baseball", "tennis", etc...

- Create a blank list called game_keys.
- Use a for loop to loop through the keys of game_dict:
    - Append the current key in the dictionary to the list game_keys.

- Create an active flag variable to control a while loop and set it to True.

- Use this flag to run a while loop:
    - To implement the randomness of the game, extra code will be needed that is outside the scope of basic Python.
        - Type import random as the first line of code in your program.

    - Randomly pick a key from the list game_keys and store it in a variable named game_category.
        - Generate a random integer that correspond to the indices of the list game_keys.
        - game_category = game_keys[?????]
    - Randomly pick a value from the dictionary that corresponds to that key and store it in a variable named game_word.
        - Generate a random integer that corresponds to the indices of the list associated as the value to the chosen key from above.
        - game_word = game_dict[game_category][?????]

    - Create an empty list called blank_word.
    - For every letter that appears in game_word:
        - Append a dash ‘-’ to blank_word

        - For instance if the game_category was fruits and the game_word was apple, you should append five "-"to the list.

    - Print a clue statement informing the user what category of words they are to guess from and how many letters there are in the word as formatted below.

    - Create an empty string called guess that will eventually hold the users guess.
    - Create a variable guess_count and set it equal to zero.

    - Use a second while loop nested inside the first to allow the user to continue guessing as long as their guess is not equal to the word they are trying to guess, which would imply they won the game. This second loop will simulate playing a single round of the game while the first loop will control playing the game or quitting.
        - Print blank_word, which is a list, as a string using the .join() method.
        - Get user input for their guess and update the value of the variable guess.
        - Increment guess_count by 1.

        - If the guess is correct:
            - Inform the user they won the game and in how many tries.
            - End the game by breaking out of the second while loop.
        - Else:
            - The guess is not correct.
            - Inform the user and reveal a letter at random to the user.
                - Make sure to reveal a different letter each time.
                - You should check that the letter you are choosing to reveal is currently represented by a "-" in the blank_word list.
                - If it is, to reveal it, replace the "-" with the corresponding letter such that when you print blank_word it will be a combination of "-" and letters.
                - If it is not, continue to pick a random letter until you find one that is.
                - To do this, create an active flag variable to control a while loop and set it to True.
                - Use this flag to run a while loop:
                    - Create a variable letter_index and set it equal to a random integer that corresponds with a random index of your list blank_word.
                    - If blank_word[letter_index] is equal to a "-":
                        - Set blank_word[letter_index] equal to the actual letter of the game_word.
                        - Set your flag variable to False.

    - Once a game round is over, get user input for if they would like to play again.
    - If the user does not want to play again:

    - Set your flag variable to False
    - Print a goodbye message thanking the user.

- Use at least 2 comments to describe sections of your code.
- “Chunk” your code so that is readable.
- Use appropriate and informative variable names.
- Format your output as below.

Example Output:

```
Welcome to the Guess My Word App

Guess a 7 letter word from the following category: Classes
-------

Enter your guess:  history
That is not correct.  Let us reveal a letter to help you!
---l---

Enter your guess:  science
That is not correct.  Let us reveal a letter to help you!
---li--

Enter your guess:  english

Correct! You guessed the word in 3 guesses.
Would you like to play again (y/n):  y


Guess a 6 letter word from the following category: Fruits
------

Enter your guess:  apples
That is not correct.  Let us reveal a letter to help you!
---a--

Enter your guess:  orange
That is not correct.  Let us reveal a letter to help you!
---a-a

Enter your guess:  bananas
That is not correct.  Let us reveal a letter to help you!
-a-a-a

Enter your guess:  banana
```

```
Correct! You guessed the word in 4 guesses.
Would you like to play again (y/n):  n


Thank you for playing our game.
```