Hangman Game Project

Project Description:

In this project, you will create a text-based Hangman game using Python. Hangman is a classic word-guessing game where the player tries to guess a word by suggesting letters within a certain number of guesses. This project will help you practice fundamental Python concepts such as loops, conditionals, functions, and string manipulation.

Project Requirements:

1. **Word List**:
    - Create a list of words from which the game will randomly choose a word for the player to guess.
2. **Game Logic**:
    - The computer selects a random word from the word list.
    - The player is prompted to guess letters one at a time.
    - The game should display the word with underscores for each letter not yet guessed (e.g., "_ _ _ _" for a 4-letter word).
    - The game should keep track of the letters the player has guessed.
    - The player has a limited number of incorrect guesses (e.g., 6). If the player reaches this limit, they lose the game.
    - If the player correctly guesses all the letters in the word before reaching the limit of incorrect guesses, they win the game.
3. **User Interface**:
    - The game should display the current state of the word (with guessed letters filled in and unguessed letters as underscores).
    - The game should display the letters the player has already guessed.
    - The game should display the number of incorrect guesses remaining.

4. **Ending the Game**:
    - The game should end when the player either guesses the word correctly or runs out of incorrect guesses.
    - At the end of the game, display a message indicating whether the player won or lost, and reveal the word.
5. **Code Structure**:
    - Use functions to organize your code. For example, you might have functions for displaying the current state of the word, checking the player's guess, and updating the game state.
    - Use clear and descriptive variable names.

Suggested Steps:

1. **Set Up the Word List**:
    - Create a list of words. You can start with a small list and expand it later.
2. **Select a Random Word**:
    - Use the `random` module to select a random word from the word list.
3. **Create the Game Loop**:
    - Implement the main game loop that will run until the player wins or loses.
4. **Handle Player Guesses**:
    - Prompt the player to guess a letter.
    - Check if the guessed letter is in the word and update the game state accordingly.
    - Validate user input.
5. **Display the Game State**:
    - Create functions to display the current state of the word, the letters guessed, and the number of incorrect guesses remaining.
6. **End the Game**:

- Determine when the game should end and display an appropriate message.

Additional Challenges (Choose one):

- Allow the player to choose the difficulty level, which could determine the length of the word or the number of incorrect guesses allowed.
- Add a feature to keep track of and display the player's score over multiple games. You can use a text file to store additional information.

Sample Demo Output:

```
Welcome to Hangman!

Current word: _ _ _ _ _ _
Guessed letters:
Incorrect guesses remaining: 6
Guess a letter: a
Sorry, 'a' is not in the word.

Current word: _ _ _ _ _ _
Guessed letters: a
Incorrect guesses remaining: 5
Guess a letter: e
Sorry, 'e' is not in the word.

Current word: _ _ _ _ _ _
Guessed letters: a, e
Incorrect guesses remaining: 4
Guess a letter: o
Good job! 'o' is in the word.

Current word: _ _ _ _ o _
Guessed letters: a, e, o
Incorrect guesses remaining: 4
Guess a letter: r
```

```
Sorry, 'r' is not in the word.

Current word: _ _ _ _ o _
Guessed letters: a, e, o, r
Incorrect guesses remaining: 3
Guess a letter: t
Good job! 't' is in the word.

Current word: _ _ t _ o _
Guessed letters: a, e, o, r, t
Incorrect guesses remaining: 3
Guess a letter: p
Good job! 'p' is in the word.

Current word: p _ t _ o _
Guessed letters: a, e, o, p, r, t
Incorrect guesses remaining: 3
Guess a letter: h
Good job! 'h' is in the word.

Current word: p _ t h o _
Guessed letters: a, e, h, o, p, r, t
Incorrect guesses remaining: 3
Guess a letter: y
Good job! 'y' is in the word.

Current word: p y t h o _
Guessed letters: a, e, h, o, p, r, t, y
Incorrect guesses remaining: 3
Guess a letter: n
Good job! 'n' is in the word.

Congratulations! You guessed the word: python
```

```
# When user fails, output the following
# Game over! The word was: python
```

Deliverables:

1. A Python script (`hangman.py`) implementing the Hangman game in your Github repository.
2. A presentation (5-10 slides) explaining your code, the logic behind it, and any challenges you faced.

Evaluation Criteria:

- **Correctness**: The game functions as described and handles edge cases (e.g., repeated guesses).
- **Code Quality**: The code is well-organized, uses functions appropriately, and includes comments where necessary.
- **User Experience**: The game provides clear instructions and feedback to the player.

Good luck, and have fun coding your Hangman game!