WORD GUESSING GAME

MINOR PROJECT

Submitted by: VAISHNAVI R

PROJECT DESCRIPTION:

The "Word Guessing Game" is a simple GUI-based game developed using Tkinter in Python. The objective of the game is to guess a randomly selected word from one of the predefined categories. The player has a limited number of guesses to correctly identify the word.

MAIN FEATURES:

- **Categories:** The game includes different categories such as fruits, programming languages, and animals from which a word is randomly selected.
- **Guessing Mechanism:** Players can enter their guesses in an input field provided.
- **Limited Guesses:** Players have a limited number of guesses to correctly identify the word.
- **Hint Functionality:** Players can request a hint which provides information about the category and length of the word.
- **Restart Option:** Players can restart the game to select a new word from a different category.

GOALS AND OBJECTIVES:

- **Entertainment:** Provide an enjoyable and engaging gaming experience for players of all ages.

- **Skill Development**: Foster problem-solving skills by challenging players to deduce the correct word within a limited number of guesses.

- **Randomization**: Utilize random selection to dynamically generate word categories and words for each game session, increasing replay value and excitement.

- **Feedback**: Provide clear and informative feedback to players regarding the correctness of their guesses, remaining guesses, and game outcomes.

- **Hint System**: Implement a hint system to assist players by providing contextual clues about the word's category and length.

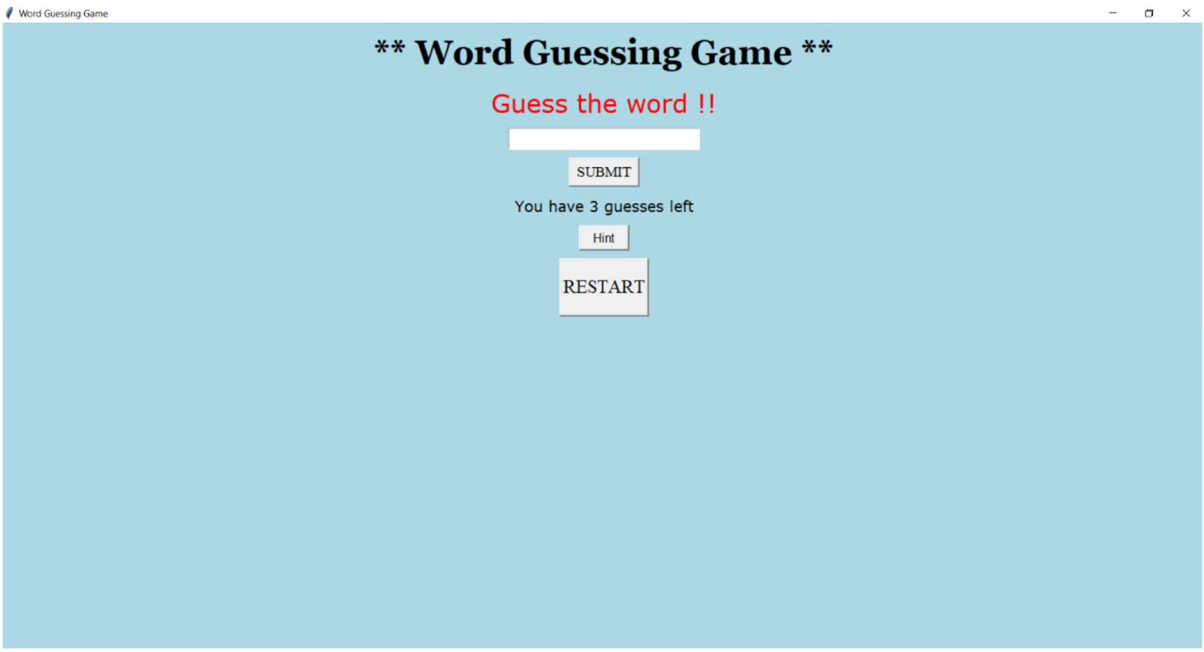
SOURCE CODE:

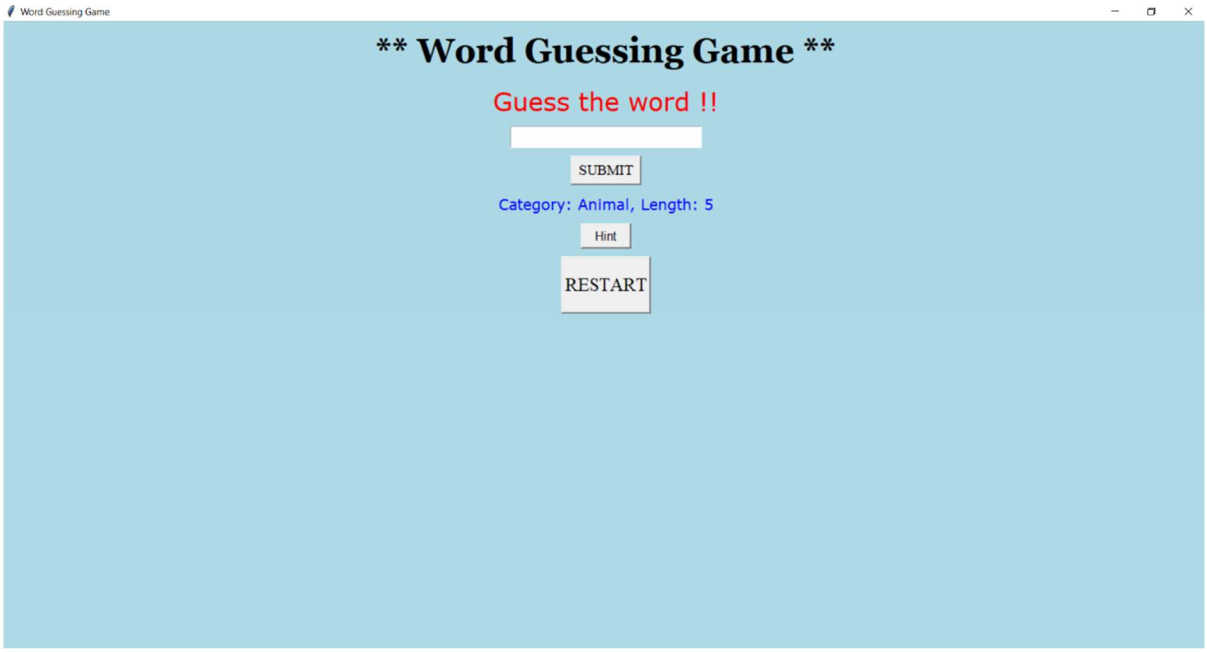
File Edit Format Run Options Window Help

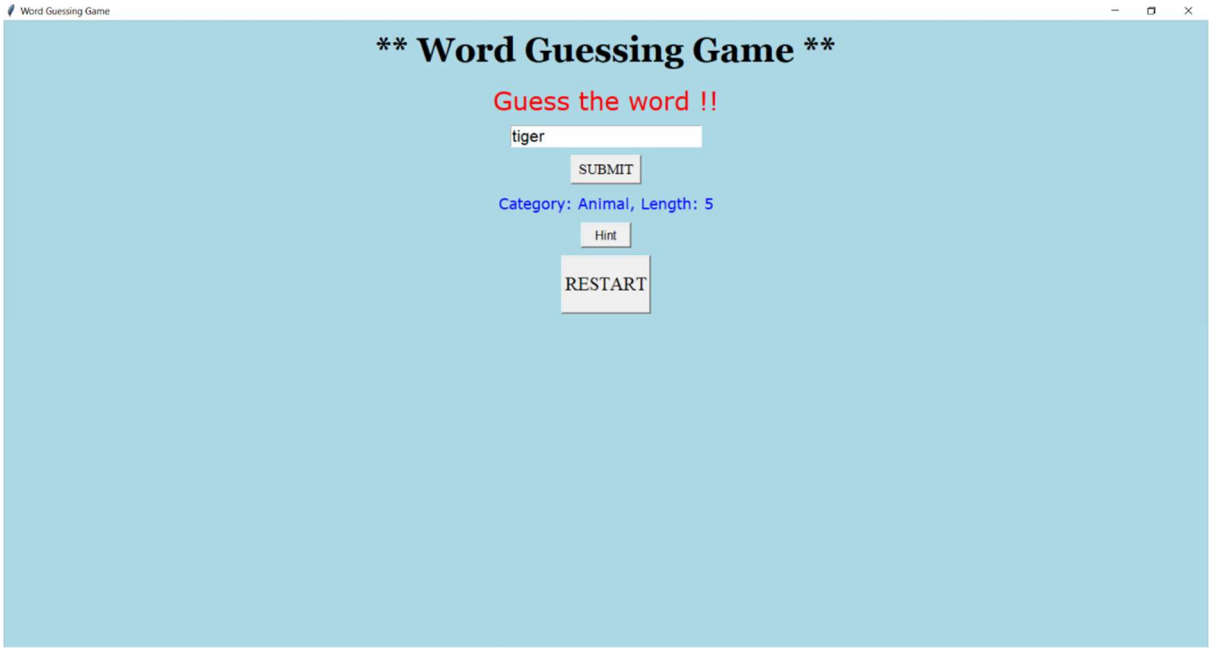
```
import tkinter as tk
import random

class WordGuessingGame:
    def __init__(self, master):
        self.master = master
        self.master.title("Word Guessing Game")
        self.master.configure(bg="#add8e6")  # Set background color to light blue
        self.words = {"fruit": ["apple", "banana", "orange", "grape", "kiwi","pineapple", "strawberry", "watermelon", "blueberry", "peach","mango", "pear", "cherry", "plum", "apricot",
                                "papaya", "melon", "lemon", "lime", "coconut"],
                      "programming": ["python", "java", "javascript", "cplusplus", "html", "css", "php"],
                      "animal": ["elephant", "lion", "tiger", "giraffe", "zebra","hippopotamus", "rhinoceros", "crocodile", "kangaroo", "koala","panda", "cheetah", "wolf", "bear", "monkey",
                                 "camel",
                                 "fox", "deer", "rabbit", "penguin"]}
        self.category = random.choice(list(self.words.keys()))
        self.secret_word = random.choice(self.words[self.category])
        self.guesses_left = 3
        self.create_widgets()

    def create_widgets(self):
        self.label_heading = tk.Label(self.master, text="** Word Guessing Game **", font=("Georgia", 32, "bold"), bg="#add8e6")
        self.label_heading.pack(pady=10)

        self.label_word = tk.Label(self.master, text="Guess the word !!", font=("Verdana", 24), bg="#add8e6",fg="red")
        self.label_word.pack(pady=5)

        self.entry = tk.Entry(self.master, font=("Arial", 16),width=20)
        self.entry.pack(pady=5)

        self.submit_button = tk.Button(self.master, text="SUBMIT", font=("Times New Roman", 14), command=self.check_guess,height=1,width=8)
        self.submit_button.pack(pady=5)

        self.result_label = tk.Label(self.master, text=f"You have {self.guesses_left} guesses left", font=("Verdana", 14), bg="#add8e6")
        self.result_label.pack(pady=5)

        self.hint_button = tk.Button(self.master, text="Hint", font=("Arial", 12), command=self.show_hint,height=1,width=6)
        self.hint_button.pack(pady=5)

        self.restart_button = tk.Button(self.master, text="RESTART", font=("Times New Roman", 18), command=self.restart_game,height=2,width=8)
        self.restart_button.pack(pady=5)
```

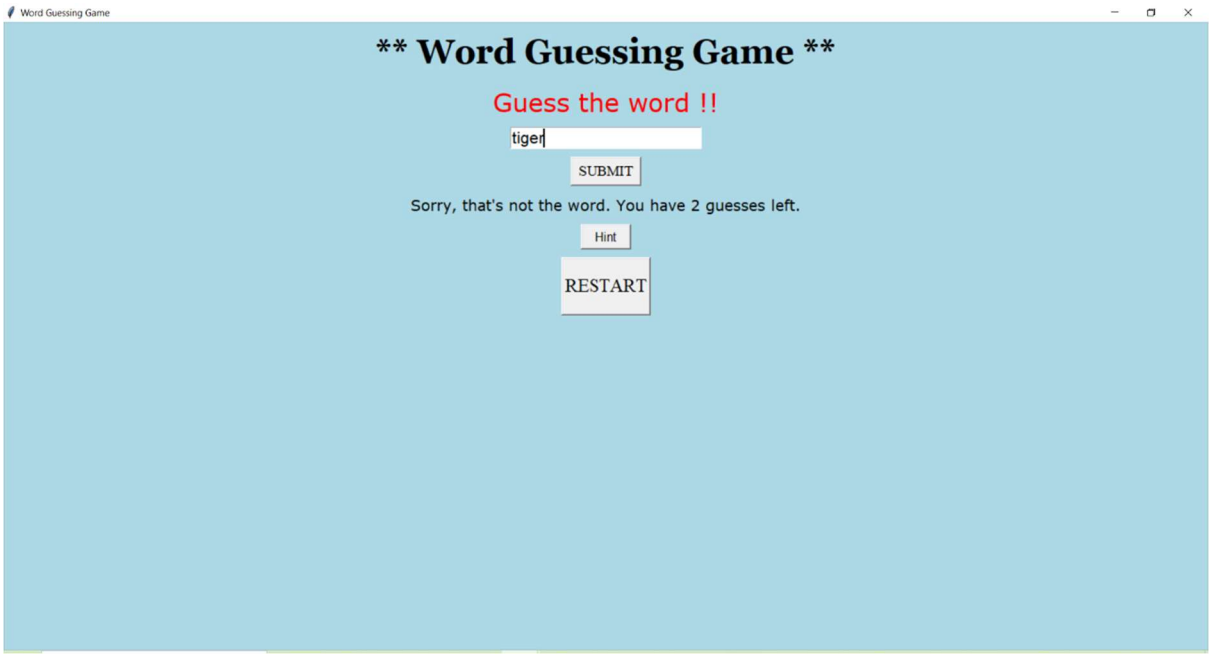
```
    def check_guess(self):
        guess = self.entry.get().lower()
        if guess == self.secret_word:
            self.result_label.config(text=f"Congratulations! You guessed the word '{self.secret_word}' correctly!", fg="green")
            self.submit_button.config(state="disabled")
            self.entry.config(state="disabled")
        else:
            self.guesses_left -= 1
            if self.guesses_left == 0:
                self.result_label.config(text=f"Sorry, you're out of guesses. The word was '{self.secret_word}'.", fg="red")
                self.submit_button.config(state="disabled")
                self.entry.config(state="disabled")
            else:
                self.result_label.config(text=f"Sorry, that's not the word. You have {self.guesses_left} guesses left.", fg="black")

    def show_hint(self):
        hint = f"Category: {self.category.capitalize()}, Length: {len(self.secret_word)}"
        self.result_label.config(text=hint, fg="blue")

    def restart_game(self):
        self.category = random.choice(list(self.words.keys()))
        self.secret_word = random.choice(self.words[self.category])
        self.guesses_left = 3
        self.label_word.config(text=f"Guess the word", fg="black")
        self.result_label.config(text=f"You have {self.guesses_left} guesses left", fg="black")
        self.submit_button.config(state="normal")
        self.entry.config(state="normal")
        self.entry.delete(0, tk.END)

def main():
    root = tk.Tk()
    game = WordGuessingGame(root)
    root.mainloop()

if __name__ == "__main__":
    main()
```

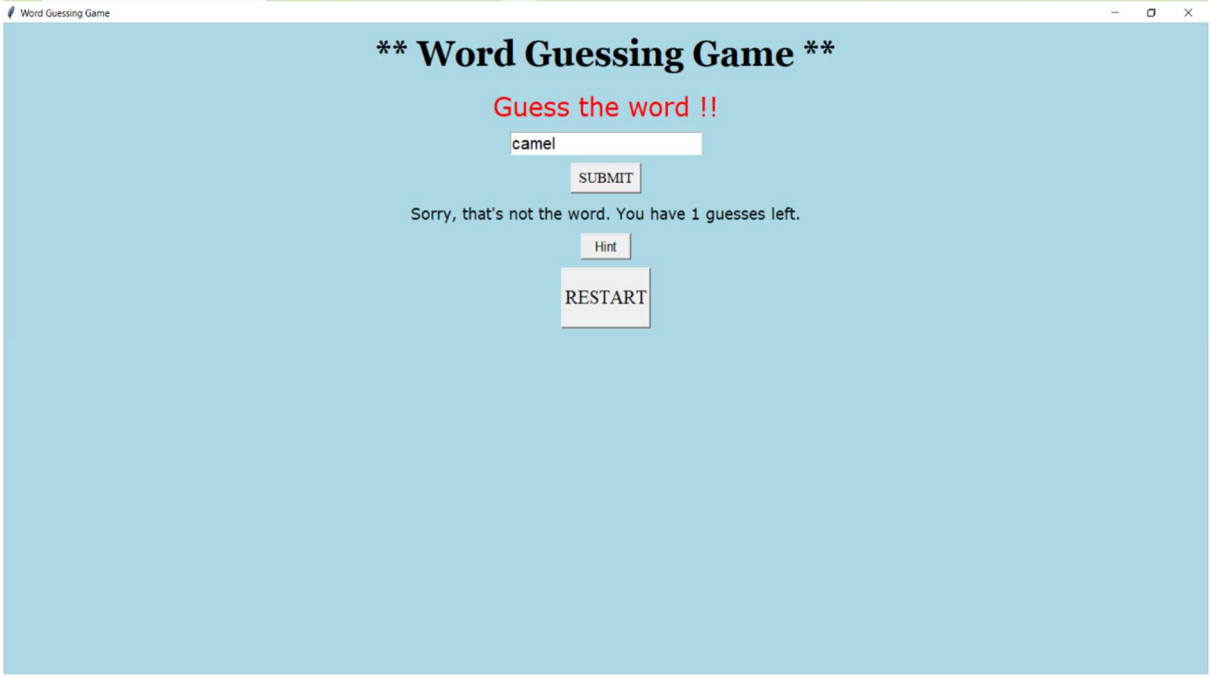
OUTPUT:

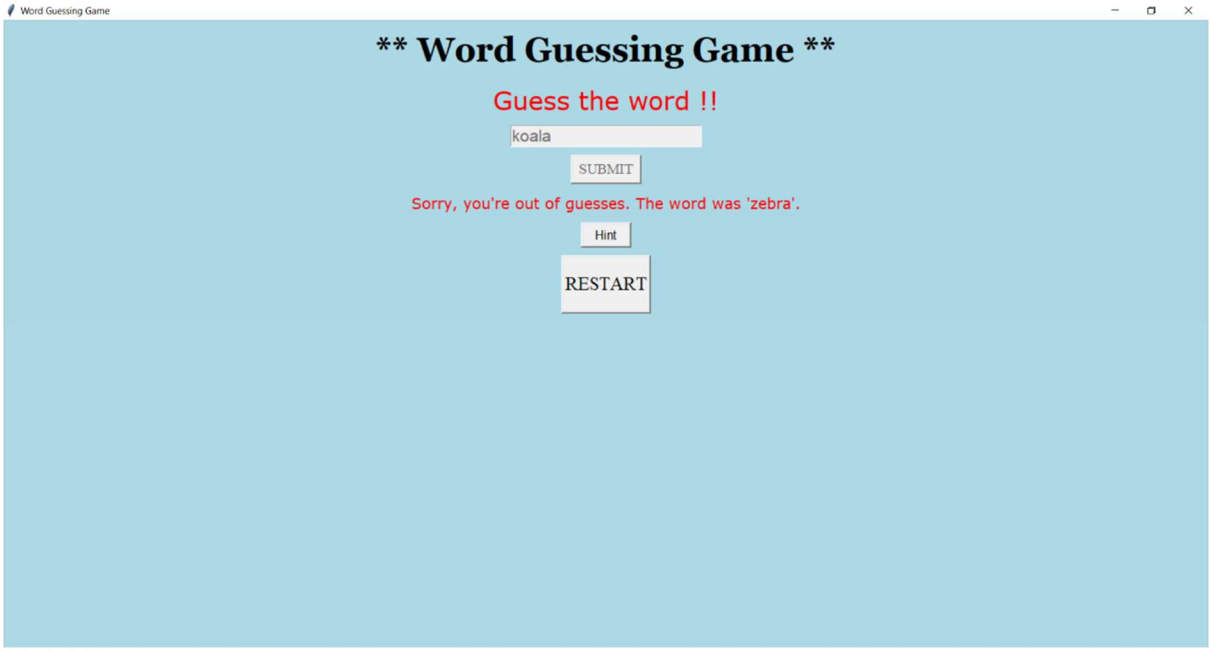


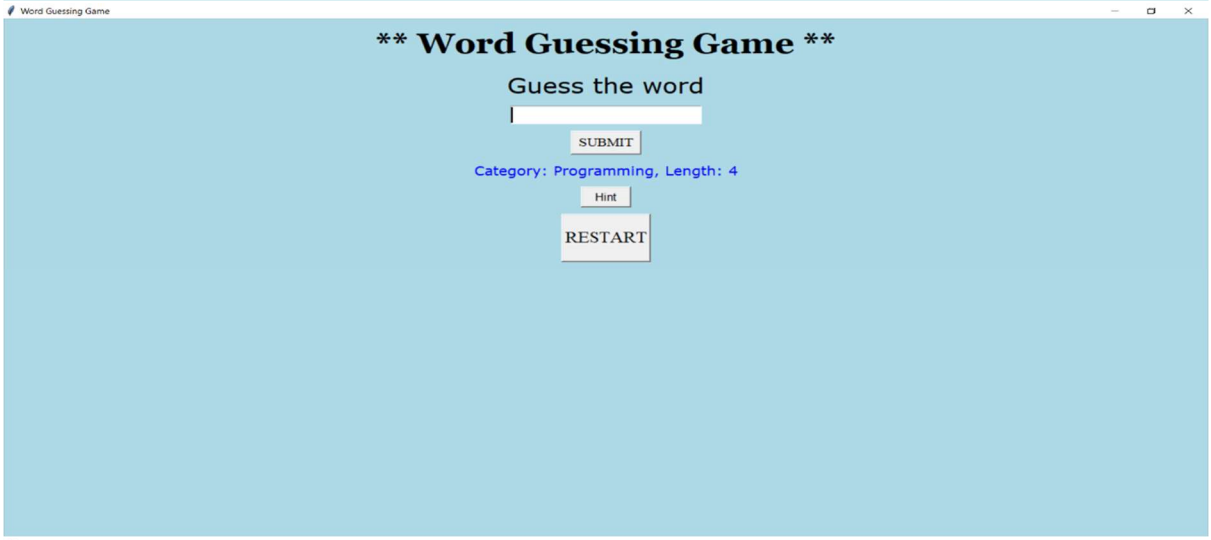

Word Guessing Game
** Word Guessing Game **
Guess the word !!
SUBMIT
Category: Animal, Length: 5
Hint
RESTART

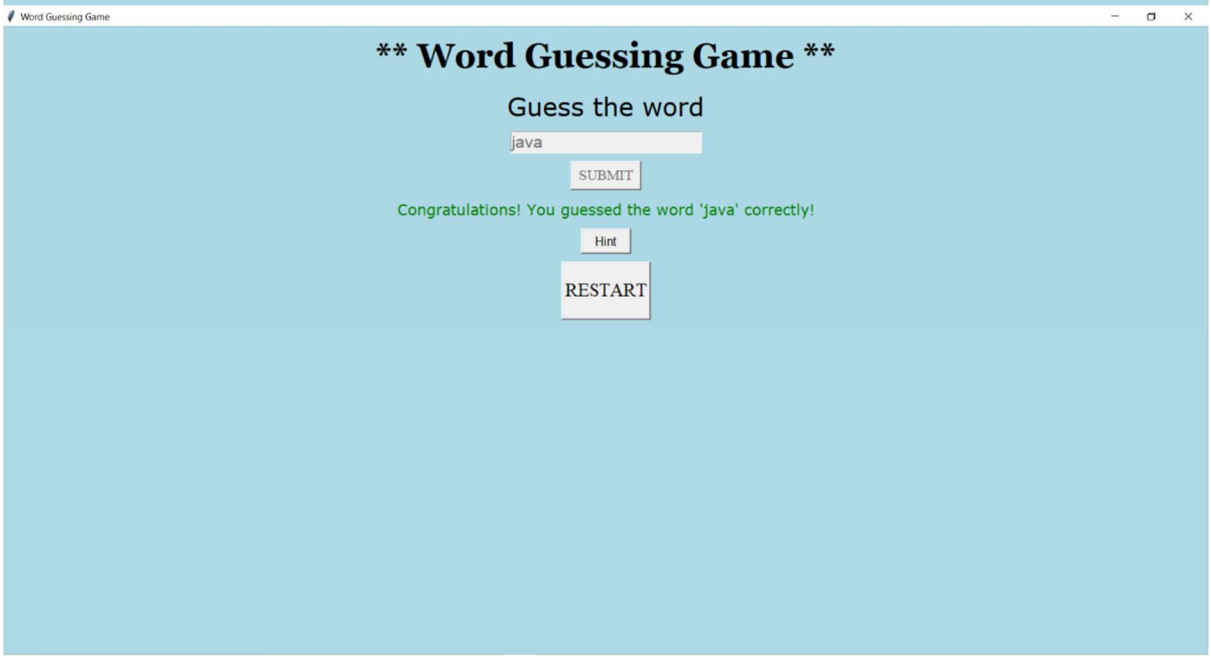

Word Guessing Game
** Word Guessing Game **
Guess the word !!
tiger
SUBMIT
Category: Animal, Length: 5
Hint
RESTART


Word Guessing Game
** Word Guessing Game **
Guess the word !!
tiger
SUBMIT
Sorry, that's not the word. You have 2 guesses left.
Hint
RESTART


Word Guessing Game
** Word Guessing Game **
Guess the word !!
camel
SUBMIT
Sorry, that's not the word. You have 1 guesses left.
Hint
RESTART


Word Guessing Game
** Word Guessing Game **
Guess the word !!
koala
SUBMIT
Sorry, you're out of guesses. The word was 'zebra'.
Hint
RESTART


Word Guessing Game
** Word Guessing Game **
Guess the word
SUBMIT
Category: Programming, Length: 4
Hint
RESTART


Word Guessing Game
** Word Guessing Game **
Guess the word
java
SUBMIT
Congratulations! You guessed the word 'java' correctly!
Hint
RESTART

RESULTS:

- Players are presented with a visually appealing GUI that includes a large heading, input field for guesses, buttons for submitting guesses and hints, and labels for displaying game status.
- The game dynamically updates the GUI to inform the player about the remaining guesses and the correctness of their guesses.
- Players receive hints about the category and length of the word to help them make informed guesses.
- Upon correctly guessing the word or exhausting all guesses, appropriate messages are displayed to inform the player of the outcome.

CONCLUSION:

The "Word Guessing Game" provides an entertaining and interactive experience for players to test their vocabulary and deduction skills. It demonstrates the use of Tkinter for building simple GUI applications in Python. With its user-friendly interface and engaging gameplay, the game offers an enjoyable pastime for players of all ages. Additionally, the game can be further enhanced with features such as scoring, difficulty levels, and more diverse word categories to provide an even richer gaming experience.

FUTURE IMPROVEMENTS:

- **Scoring System**: Implement a scoring mechanism to award points based on the number of guesses taken or the time elapsed to guess the word. Players can compete for high scores and track their progress over multiple game sessions.
- **Difficulty Levels**: Introduce multiple difficulty levels (e.g., easy, medium, hard) with varying constraints such as fewer guesses for higher difficulty levels or longer words for increased challenge.

- **Multiplayer Mode**: Add multiplayer functionality to allow multiple players to compete against each other in real-time or take turns guessing words. This can be implemented locally or over a network.
- **Custom Word Lists**: Enable players to create custom word lists or import external word lists to expand the variety of words available for guessing.
- **Visual Enhancements**: Improve the visual aesthetics of the GUI by incorporating custom graphics, animations, or themes to enhance the overall look and feel of the game.