**HANGMAN GAME IMPLEMENTATION IN PYTHON**

Submitted by:

K T Sowndharyaa

B.E. CSE – 3rd year

Anna University Regional Campus Coimbatore

1. **Introduction**

The Hangman game is a classic word guessing game where players attempt to guess a word by suggesting letters within a limited number of attempts. It not only provides entertainment but also aids in vocabulary building and logical thinking. This project focuses on implementing the Hangman game in Python, a popular programming language known for its simplicity and versatility.

1. **Objective**

The primary objective of this project is to create a functional Hangman game using Python. By implementing this game, we aim to demonstrate proficiency in Python programming concepts such as classes, methods, loops, and conditional statements. Additionally, the project seeks to provide an interactive and enjoyable experience for users while reinforcing their knowledge of Python syntax and data structures.

1. **Materials and methods**

**3.1 Software requirements**

Python (version 3.0 or higher)

Integrated Development Environment (IDE) such as PyCharm, Visual Studio Code, or Jupyter Notebook.

**3.2 Implementation**

The Hangman game is implemented using object-oriented programming principles in Python. The key components of the implementation include:

A Hangman class to manage the game state and logic.

Methods for word selection, letter guessing, and game outcome determination.

Utilization of lists, strings, loops, and conditional statements to create the game functionality.

1. **Code and Results**

**Code**

import random

import os

import time

def clear\_screen():

# Clear the console screen

os.system('cls' if os.name == 'nt' else 'clear')

def print\_hangman\_stage(stage):

# Print the hangman stage

hangman = [

"""

\_\_\_\_\_\_\_\_\_

| |

| O

| \\|/

| |

| / \\

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

| O

| \\|/

| |

| /

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

| O

| \\|/

| |

|

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

| O

| \\|

| |

|

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

| O

| |

| |

|

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

| O

|

|

|

|

""",

"""

\_\_\_\_\_\_\_\_\_

| |

|

|

|

|

|

"""

]

print(hangman[stage])

def play\_hangman():

words = ['apple', 'banana', 'orange', 'grape', 'pineapple', 'kiwi', 'watermelon','mango', 'strawberry', 'blueberry','berry','grape', 'grapefruit','apricot','avocado','papaya','cherry','peach','muskmelon','custardapple','lychee','pomegranate']

word = random.choice(words)

guessed\_letters = set()

attempts = 6

success = False

print("Welcome to Hangman!")

time.sleep(1)

print("How to play?\n\tThe computer will think of a random word and keep it a secret.\n\tYou have to figure out the word by guessing one letter at a time.\n\tIf the guessed letter is in the word, it gets revealed in its correct position(s).\n\tIf the guessed letter is not in the word, a part of a stick figure (representing a person) gets drawn.\n\tYou keep guessing letters until you either figure out the word or the stick figure is fully drawn (which means you lose).\n\tThe game ends when the word is guessed correctly or when the stick figure is completed.")

input("Press Enter to continue...")

clear\_screen()

while attempts > 0:

print\_hangman\_stage(6 - attempts)

print("Word:", end=' ')

for letter in word:

if letter in guessed\_letters:

print(letter, end=' ')

else:

print('\_', end=' ')

print("\n")

if all(letter in guessed\_letters for letter in word):

success = True

break

print("Hint: The secret word may be a fruit")

print("You need to guess", len(word), "letter word")

guess = input("Guess a letter: ").lower()

if len(guess) != 1 or not guess.isalpha():

print("Please enter a single letter.")

time.sleep(1)

continue

if guess in guessed\_letters:

print("You already guessed that letter.")

time.sleep(1)

continue

guessed\_letters.add(guess)

if guess not in word:

attempts -= 1

clear\_screen()

if success:

print("Congratulations! You guessed the word:", word)

else:

print("Sorry, you ran out of attempts. The word was:", word)

# Main loop to allow restarting the game

while True:

play\_hangman()

restart = input("Do you want to play again? (yes/no): ").lower()

if restart != 'yes':

print("Thanks for playing Hangman!")

break

The Hangman game implementation successfully creates an interactive gaming experience where users can:

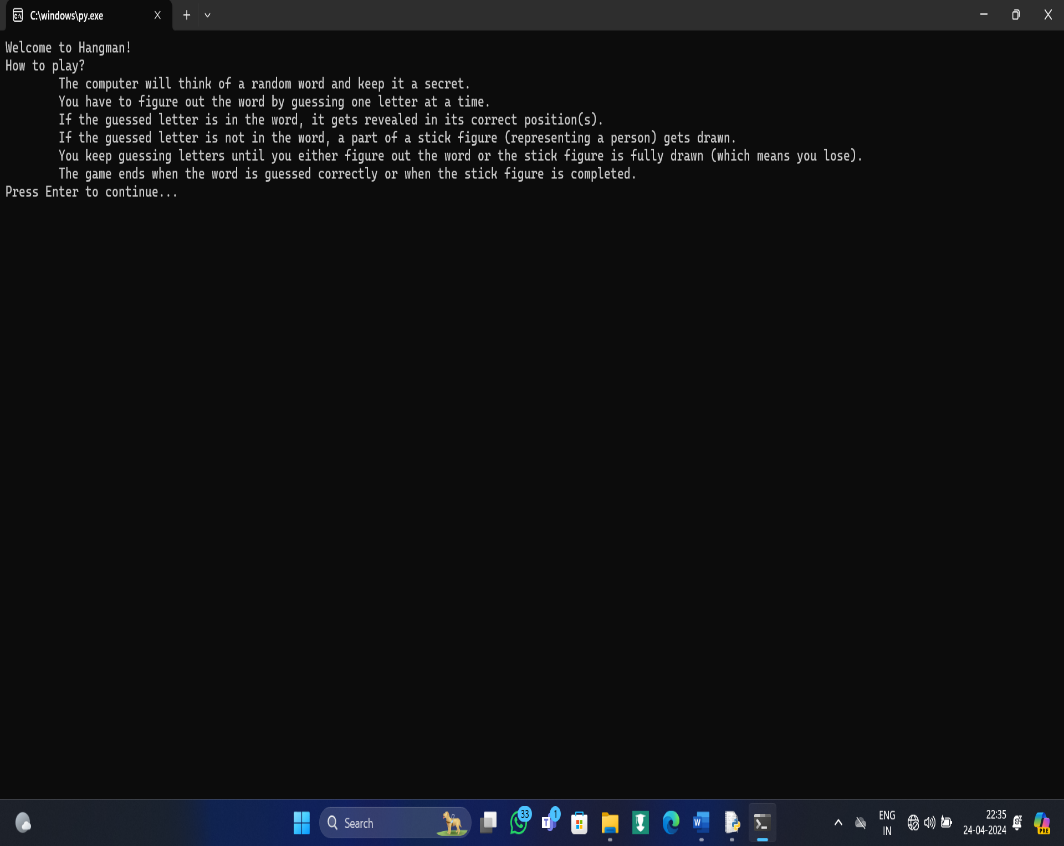
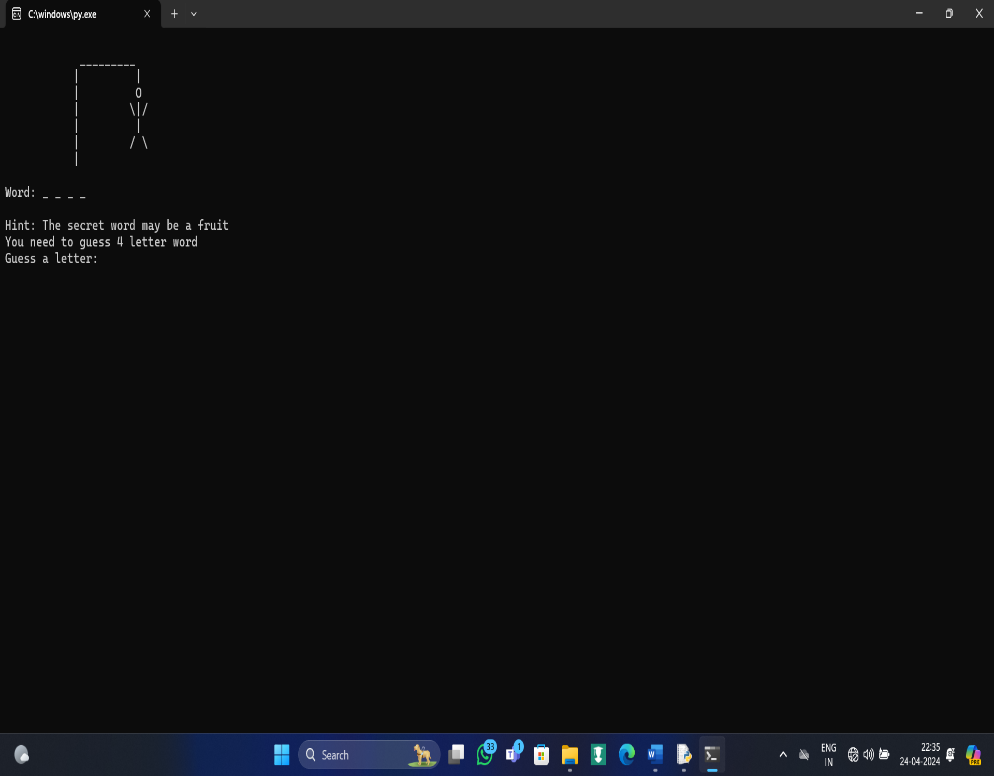
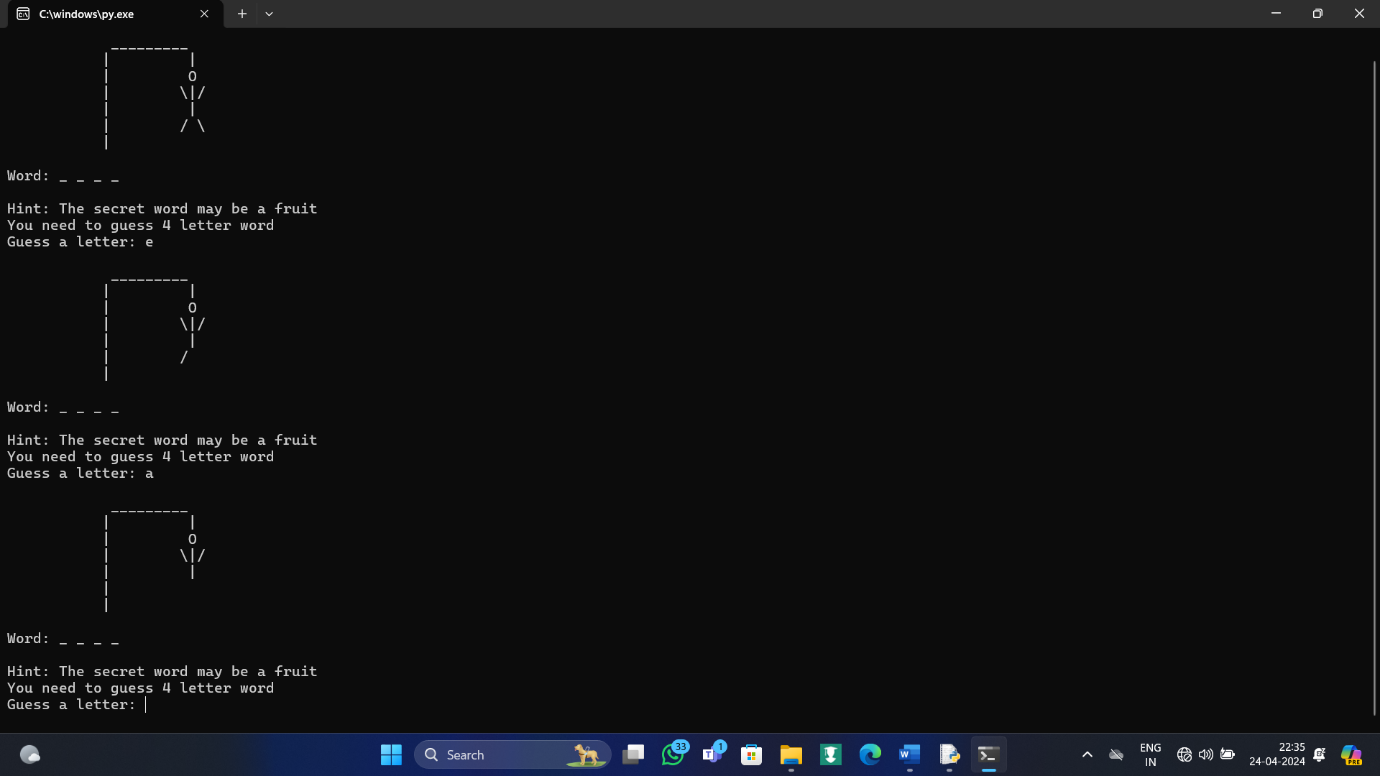
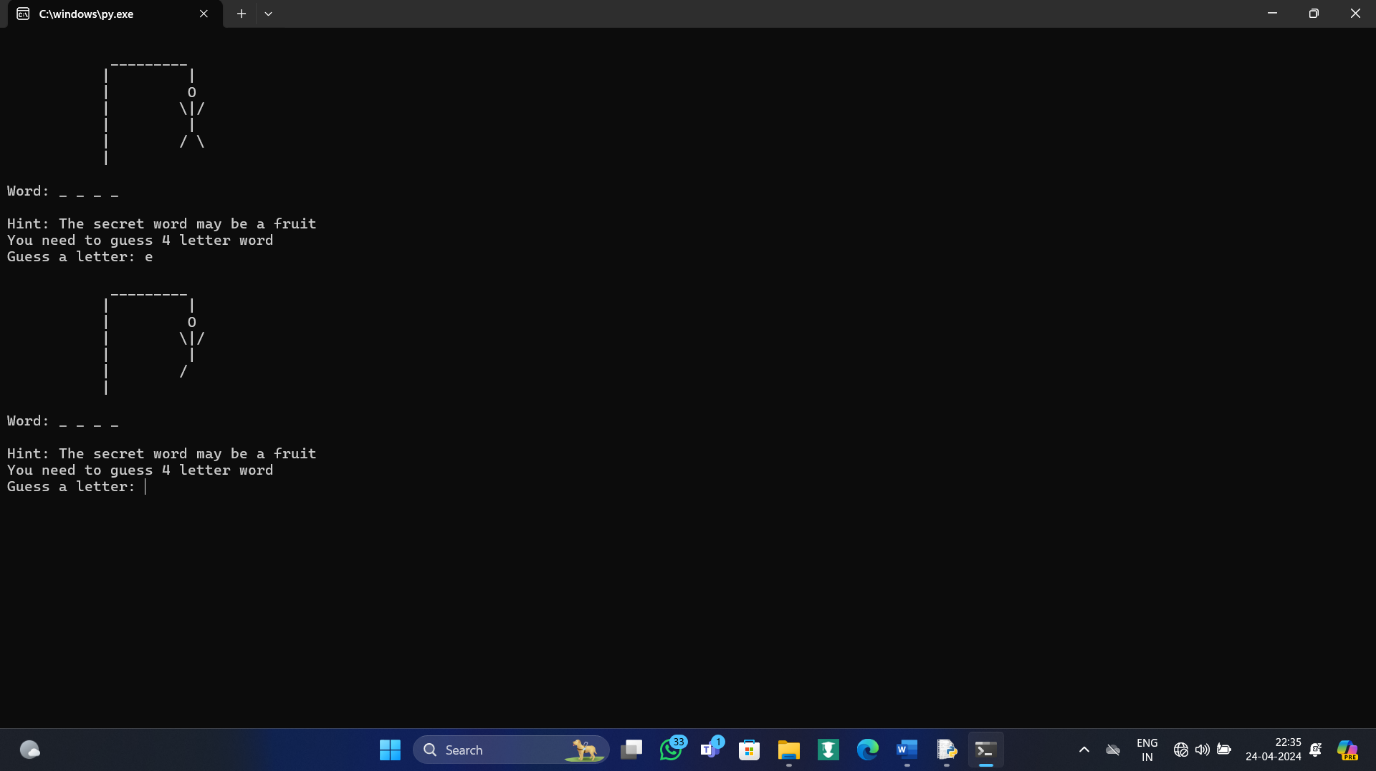
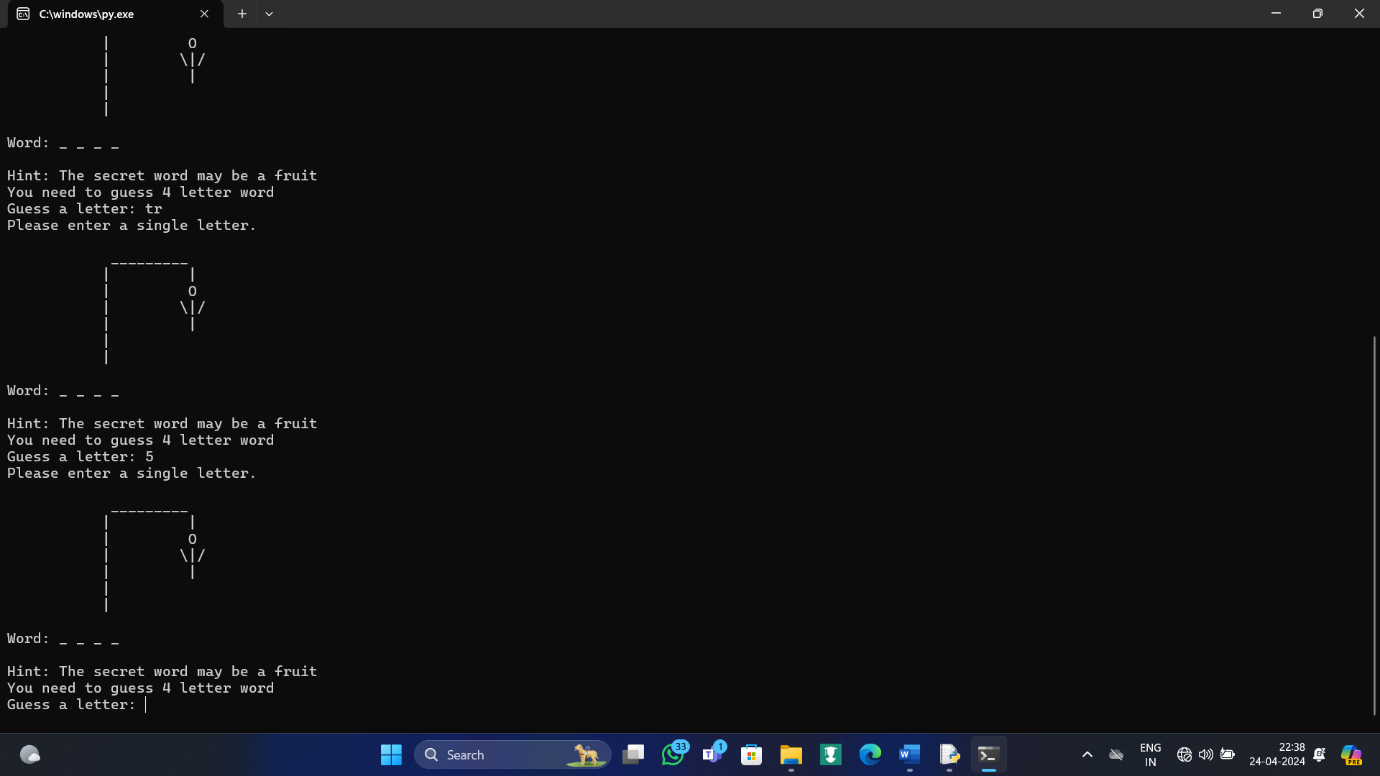
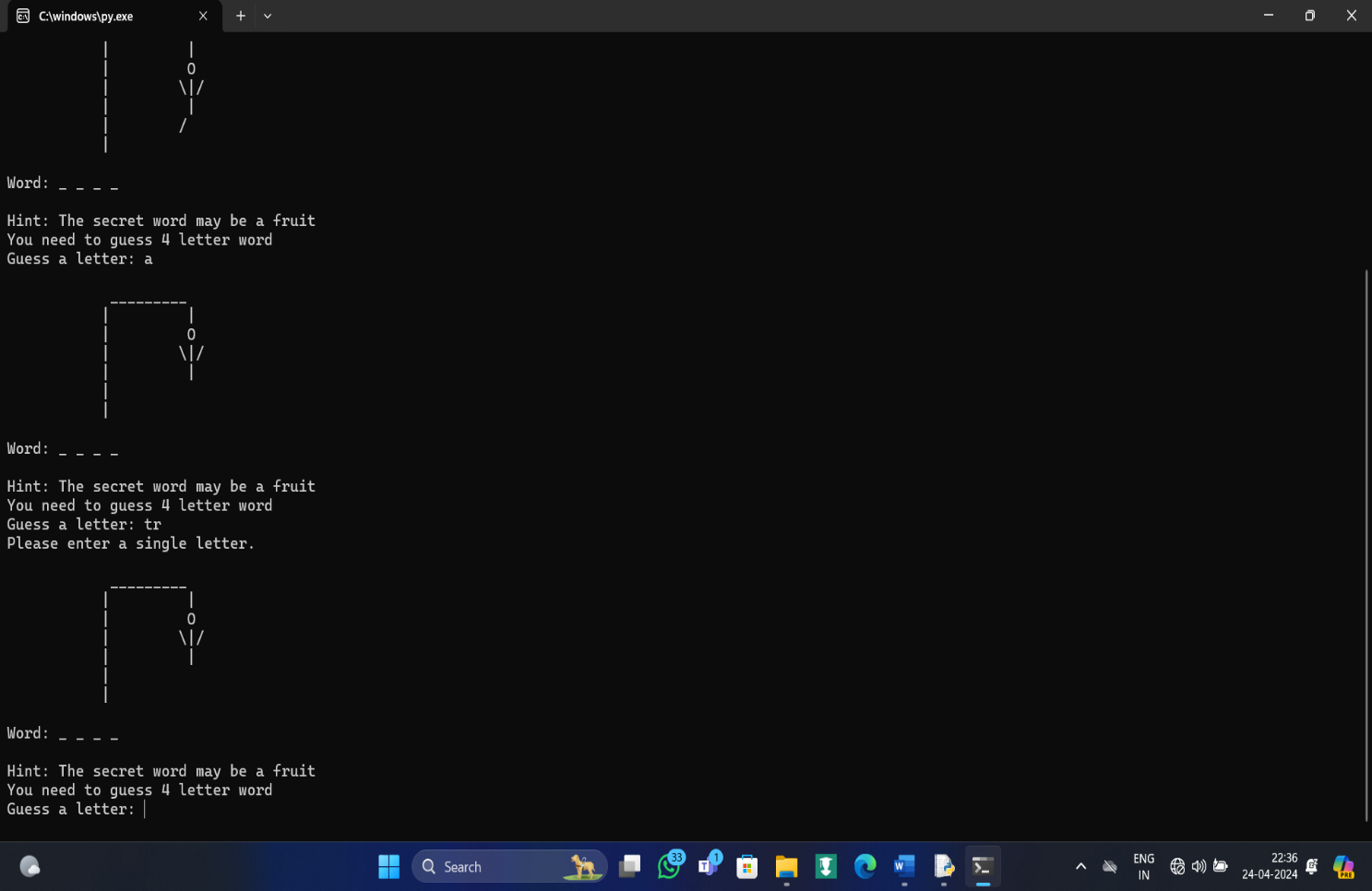
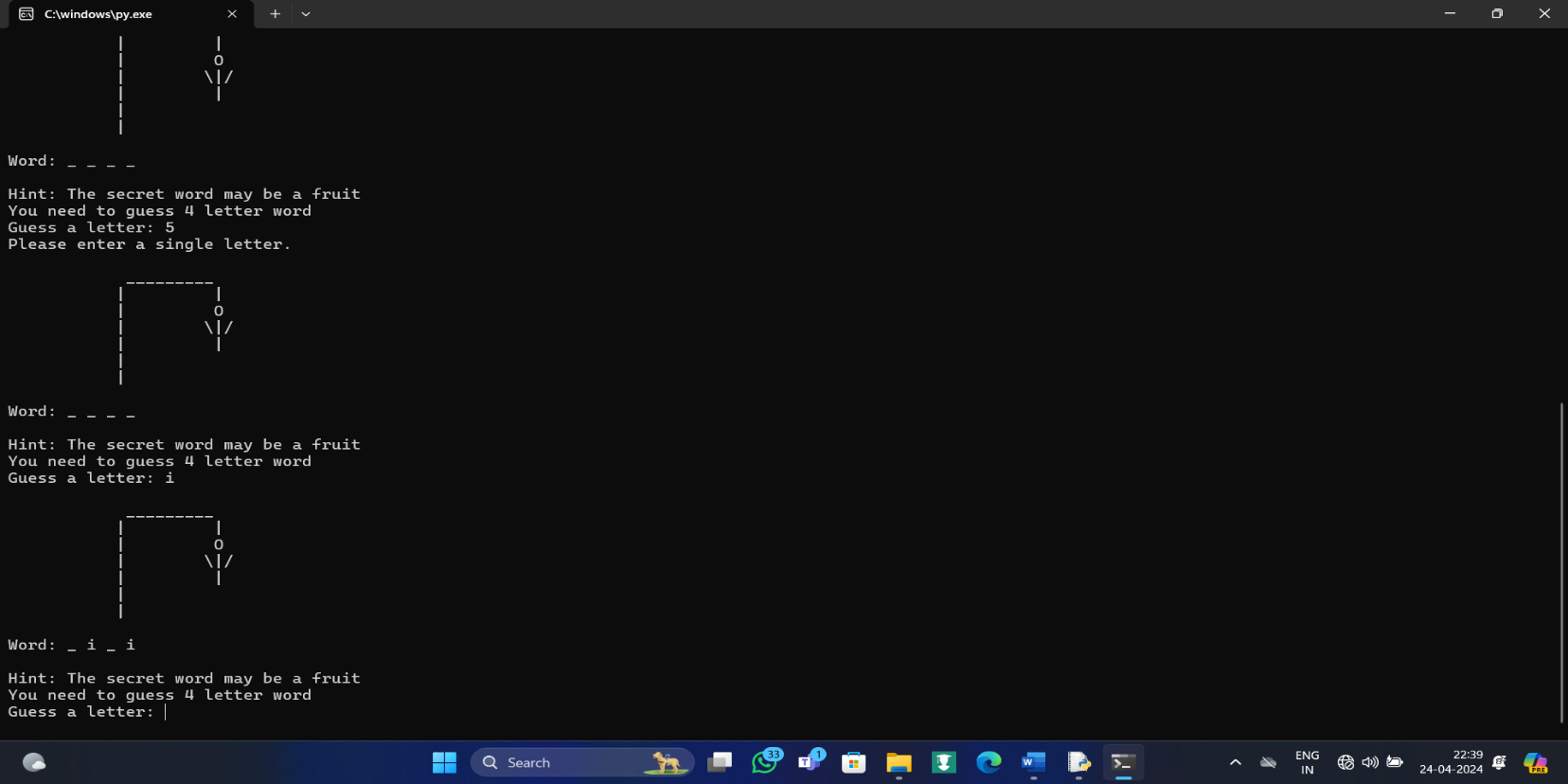
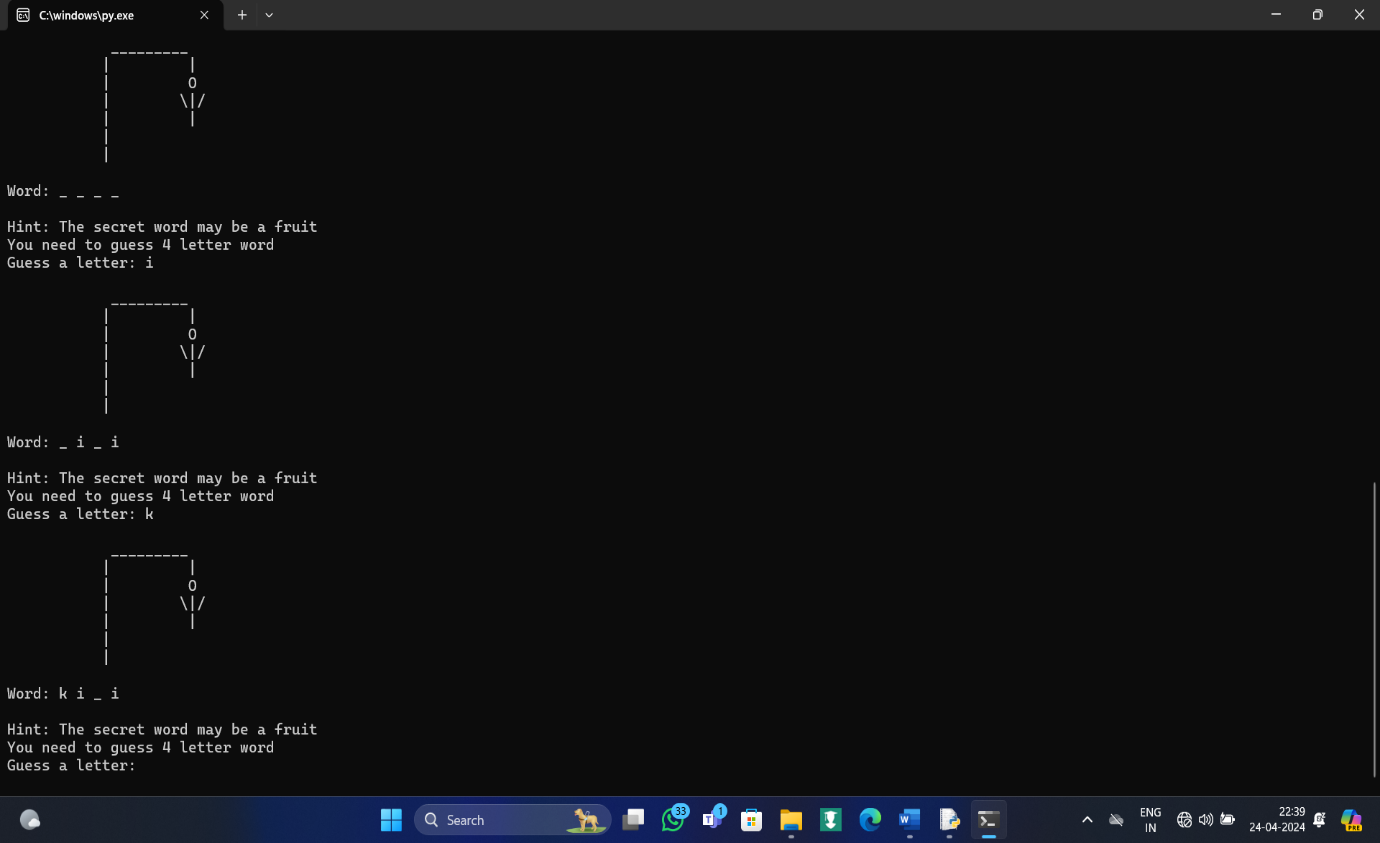
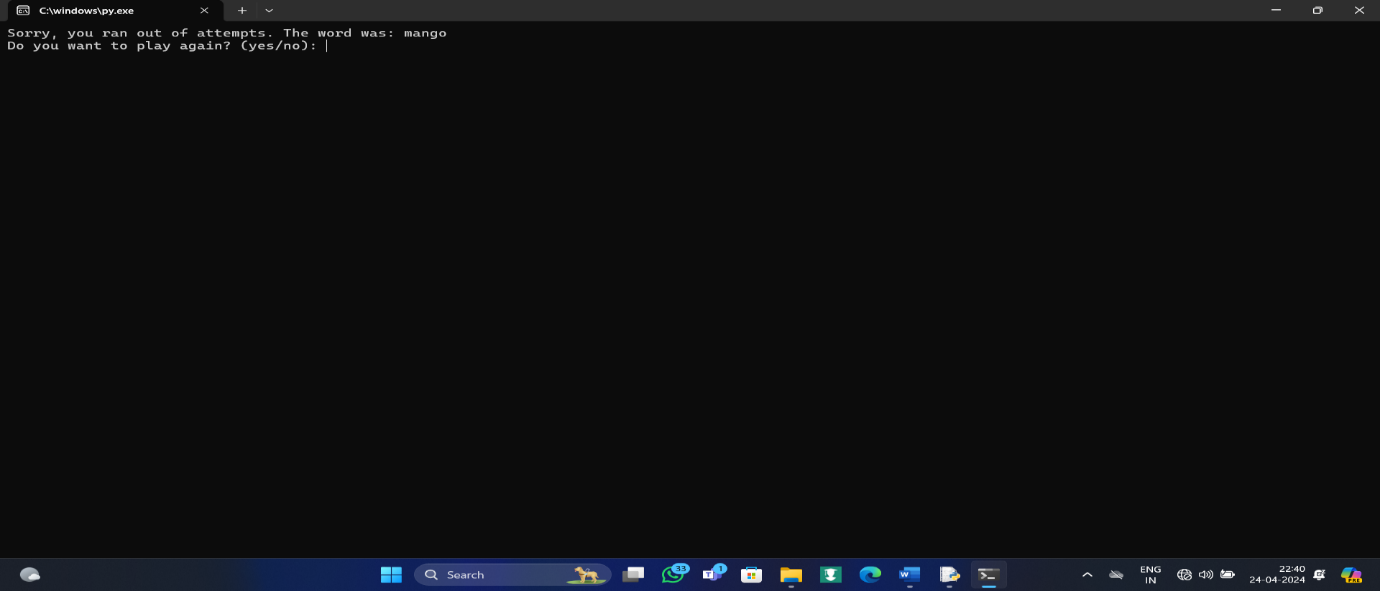
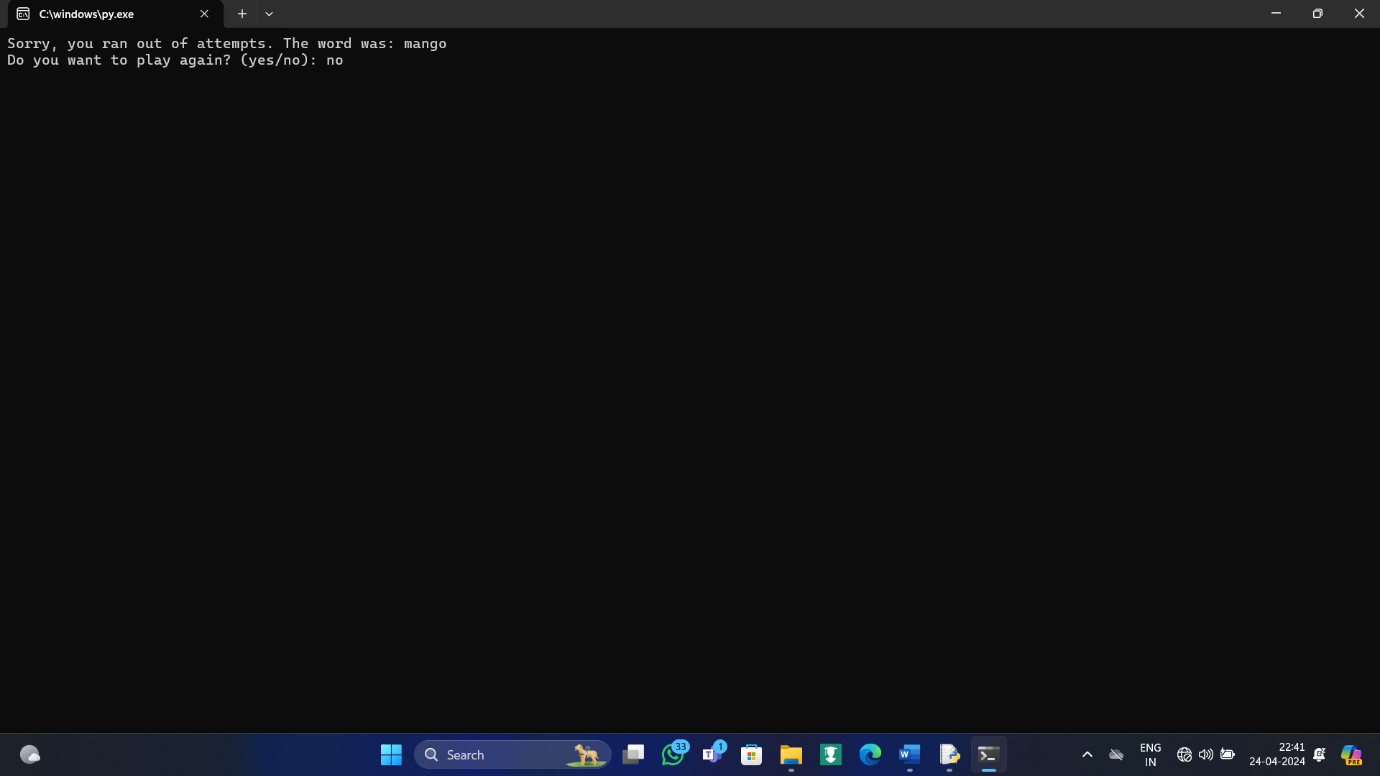
Guess letters to reveal the hidden word.

Trak their remaining attempts.

Receive feedback on their guesses and game outcome.

The functionality is demonstrated through screenshots and code snippets showcasing different stages of the game, including initial word display, guessed letters, and final game outcome.

**Result**

** **      

1. **Discussion**

The implemented Hangman game effectively simulates the traditional paper-and-pencil version, providing users with an engaging and educational experience. Challenges encountered during implementation include:

Ensuring random word selection from a predefined list.

Handling user input validation and error checking.

Despite these challenges, the game was successfully implemented and provides an entertaining platform for users to test their word-guessing skills.

1. **Conclusion**

In conclusion, the implementation of the Hangman game in Python demonstrates the versatility and practicality of the Python programming language. By combining fundamental programming concepts with interactive gameplay, the project achieves its objective of providing an entertaining and educational experience for users. Moving forward, potential enhancements to the game could include integration of a graphical user interface (GUI) or expansion of the word list to increase variety and challenge.

1. **References**

Python documentation: <https://docs.python.org/>

Stack overflow community for troubleshooting and code snippets: https://stackoverflow.com/