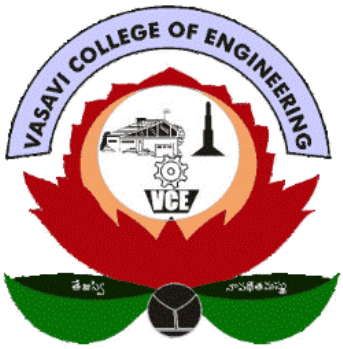
A Course Based Project
On

WORDLE - A CONSOLE-BASED WORD GAME

By
Shaziya Shaik
1602-24-748-042
G. Neethi Amrutha Pranuthi
1602-24-748-029

Course: Python Programming
Class: BE (CSE AIML Semester II)



Department of Computer Science and Engineering

Vasavi College of Engineering (Autonomous)
Ibrahimbagh, Hyderabad - 31
2024

ABSTRACT

This Python project presents a terminal-based implementation of the popular Wordle game, focusing on modular design, string manipulation, and dynamic user interaction. The program employs various Python modules to provide core functionalities such as randomized word selection from a text file and input validation. The game randomly selects a five-letter target word and challenges the user to guess it within a limited number of attempts, offering color-coded feedback using ANSI escape codes on letter accuracy and position after each guess. The project closely resembles the original Wordle game while being lightweight and extensible.

The project demonstrates practical application of fundamental programming concepts such as control structures, loops, conditionals, string operations, exception handling, and modularization. The use of external modules like random for randomness, os for terminal control, and string for character operations enhances the clarity and maintainability of the codebase. The game flow ensures robustness by validating user input against a predefined word list and handling edge cases.

Key Features:

Randomized Word Selection: Uses the random module to choose a new target word from a predefined list, ensuring varied gameplay each time.

Terminal UI Handling: The os.system('cls' or 'clear') function is used to clear the console after each guess, offering a clean and readable interface. The importing of the OS module is a good example of using modules in the program.

Modular Structure: Organized into functions such as get_random_word(), validate_input(), check_guess(), and display_feedback() to improve readability and facilitate future updates.

Introduction

This Python program is a terminal-based Wordle-style guessing game that challenges players to guess a secret 5-letter word within six attempts. Inspired by the popular Wordle game, this version includes themed word categories such as College Wordle, Films, Books, Songs, and Sports, making it more diverse in topics as compared to the original Wordle game.

The game uses the following features:

- Color-coded feedback using the `colorama` library:
    - 🟩 Green for correct letters in the correct position.
    - 🟨 Yellow for correct letters in the wrong position.
    - ⬜ Gray for letters not in the word.
- Robust input validation, ensuring guesses are exactly 5 letters and present in the word list.
- A clean and interactive experience using screen clearing and structured feedback.
- Modular code structure with functions for word loading, evaluation, validation, and gameplay logic.

By combining basic control structures, file handling, string manipulation, and user feedback, this game serves as an engaging application of fundamental Python programming concepts. Furthermore, it involves complex algorithms that select random words and ask the user to guess it, making it a fairly moderate-to-difficult word game unlike other word games.

Methodology

The Wordle-style game was developed using a modular Python approach for clarity and efficiency. The program begins by importing necessary libraries (`colorama`, `os`, and `random`) to handle color output, screen clearing, and word selection.

The user selects from five game categories (College, Films, Books, Songs, Sports), and a corresponding list of 5-letter words is loaded from a text file. The game then randomly selects a secret word and allows the player up to six attempts to guess it.

Each guess is validated for length and existence in the word list. Feedback is given using color codes:

- 🟩 Green for correct letters in the right position
- 🟨 Yellow for correct letters in the wrong position
- ⬜ Gray for incorrect letters

Functions are used to handle word loading, input checking, and feedback evaluation, promoting clean and reusable code. The game ends with a success or failure message based on the player's performance.

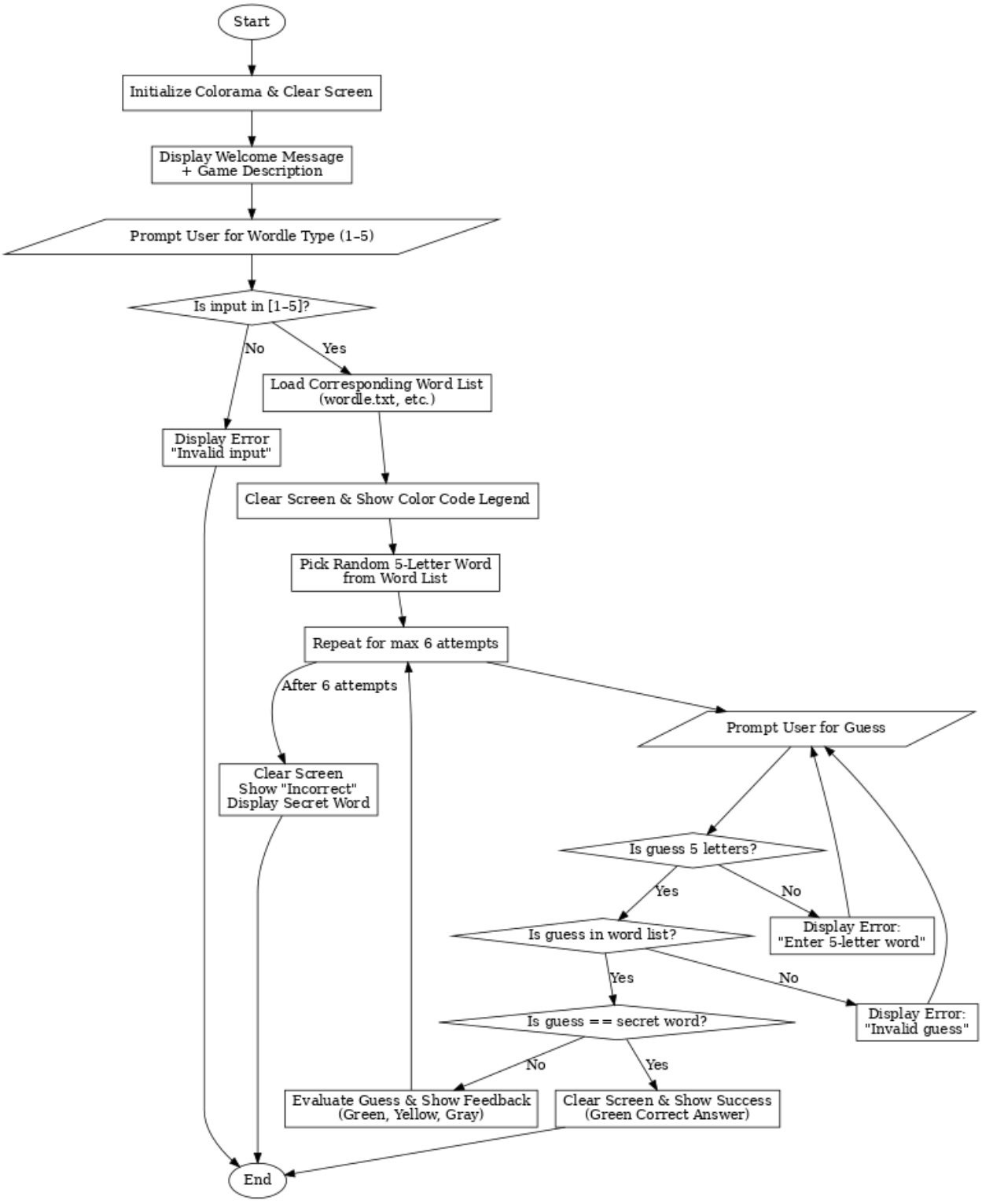
Motivation

The motivation behind creating this Wordle-style game was to blend entertainment with education while strengthening practical programming skills in Python. Inspired by the popularity of Wordle, the goal was to design a version that not only mimics its engaging gameplay but also introduces category-based word challenges to make the experience more meaningful and personalized.

Modules Used

random: For selecting a random target word.
os: For screen-clearing to maintain a neat command-line interface.
string: For validating alphabetic characters and managing character sets.

Flowchart


Start
Initialize Colorama & Clear Screen
Display Welcome Message + Game Description
Prompt User for Wordle Type (1–5)
Is input in [1–5]?
No
Yes
Load Corresponding Word List (wordle.txt, etc.)
Display Error "Invalid input"
Clear Screen & Show Color Code Legend
Pick Random 5-Letter Word from Word List
Repeat for max 6 attempts
After 6 attempts
Prompt User for Guess
Clear Screen Show "Incorrect" Display Secret Word
Is guess 5 letters?
Yes
No
Is guess in word list?
Display Error: "Enter 5-letter word"
Yes
No
Is guess == secret word?
Display Error: "Invalid guess"
No
Yes
Evaluate Guess & Show Feedback (Green, Yellow, Gray)
Clear Screen & Show Success (Green Correct Answer)
End

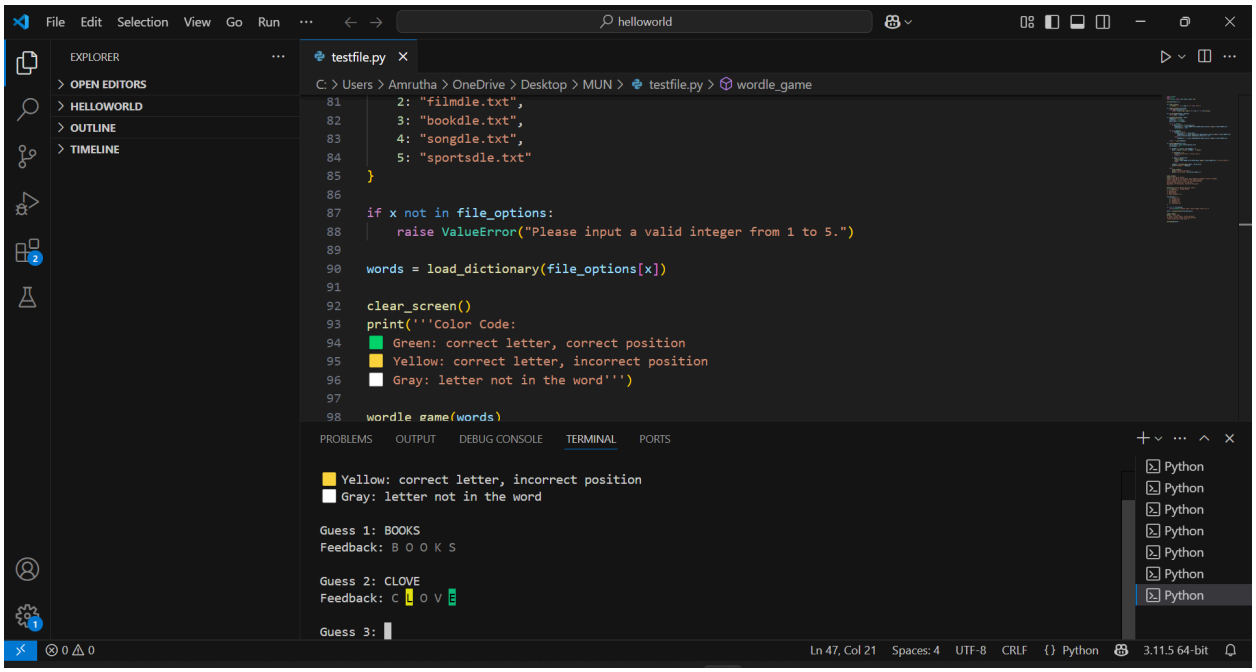
Source Code

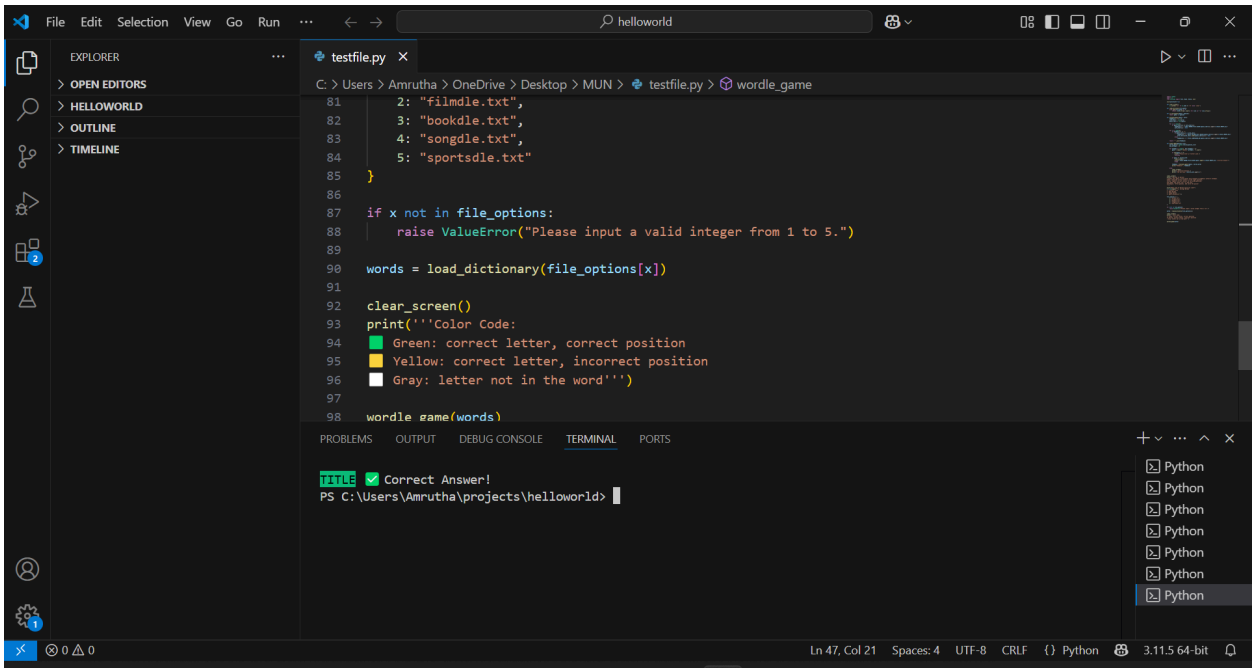
```
import random
import os
from colorama import Fore, Back, Style, init

init(autoreset=True)

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

def load_dictionary(file_path):
    with open(file_path) as f:
        return [line.strip().lower() for line in f if line.strip()]

def is_valid_guess(guess, guesses):
    return guess in guesses

def evaluate_guess(guess, word):
    feedback = [""] * 5
    word_chars = list(word)
    guess_chars = list(guess)

    for i in range(5):
        if guess_chars[i] == word_chars[i]:
            feedback[i] =
f"{Back.GREEN}{Fore.BLACK}{guess_chars[i].upper()}{Style.RESET_ALL}"
            word_chars[i] = None

    for i in range(5):
        if feedback[i] == "":
            if guess_chars[i] in word_chars:
                feedback[i] =
f"{Back.YELLOW}{Fore.BLACK}{guess_chars[i].upper()}{Style.RESET_ALL}"
                word_chars[word_chars.index(guess_chars[i])] = None
            else:
                feedback[i] =
f"{Fore.LIGHTBLACK_EX}{guess_chars[i].upper()}{Style.RESET_ALL}"

    return " ".join(feedback)
```

```
def wordle_game(guesses_list):
    secret_word = random.choice(guesses_list)
    max_attempts = 6

    for attempt in range(1, max_attempts + 1):
        guess = input(f"\nGuess {attempt}: ").lower()

        if len(guess) != 5:
            print(f"Please enter a 5-letter word.")
            continue

        if not is_valid_guess(guess, guesses_list):
            print("Invalid guess. Please enter a valid English word.")
            continue

        if guess == secret_word:
            clear_screen()
            print(f"{Back.GREEN}{Fore.BLACK}{guess.upper()}{Style.RESET_ALL} ✅ Correct
Answer!")
            break

        feedback = evaluate_guess(guess, secret_word)
        print("Feedback:", feedback)

    else:
        clear_screen()
        print("❌ Incorrect Guesses.")
        print(f"The word was: {secret_word.upper()}")


clear_screen()
print("""Welcome to Wordle!
A player must guess a five-letter word related to academics within 6 attempts.
Green: indicates correct letters in the right position,
Yellow: signals correct letters in the wrong position,
Gray means the letter isn't in the word.
Strategize, refine guesses, and solve the puzzle!
""")
```

```
print("What kind of Wordle would you like?")
x = int(input("""1. College Wordle
2. 5-Letter Films
3. Book Wordle
4. Songs Wordle
5. Sports Wordle\n"""))

file_options = {
    1: "wordle.txt",
    2: "filmdle.txt",
    3: "bookdle.txt",
    4: "songdle.txt",
    5: "sportsdle.txt"
}

if x not in file_options:
    raise ValueError("Please input a valid integer from 1 to 5.")

words = load_dictionary(file_options[x])

clear_screen()
print(‘‘‘Color Code:
        🟩 Green: correct letter, correct position
        🟨 Yellow: correct letter, incorrect position
        ⬜ Gray: letter not in the word’’’)

wordle_game(words)
```

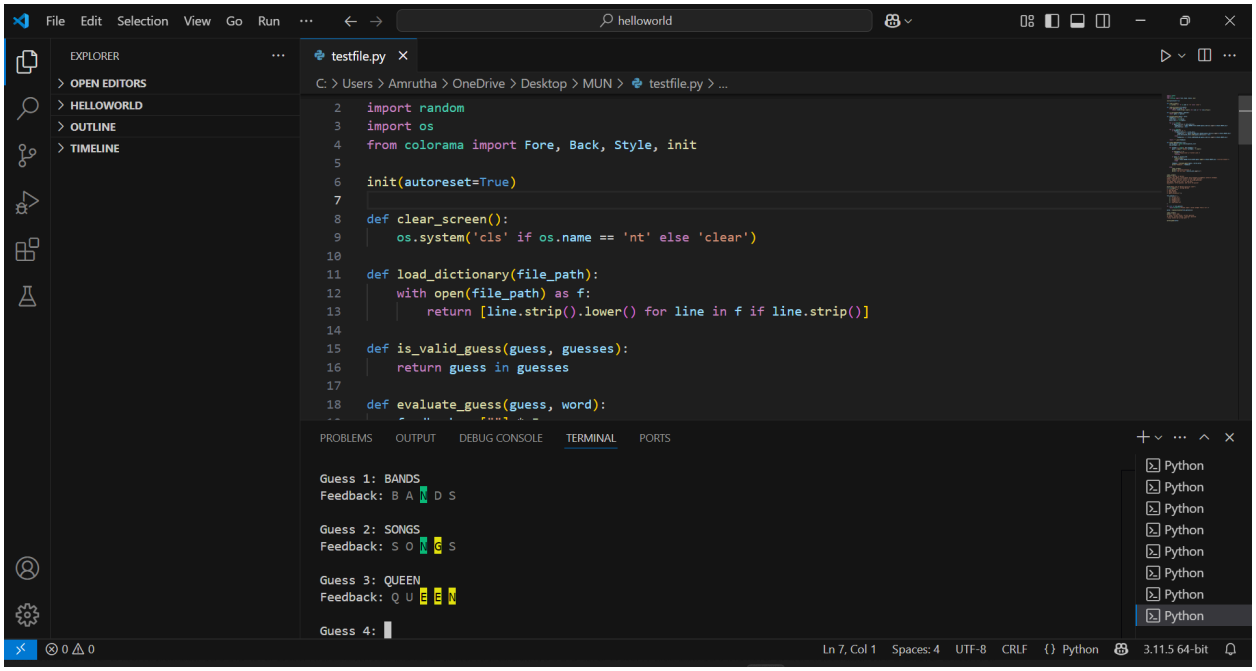
Results

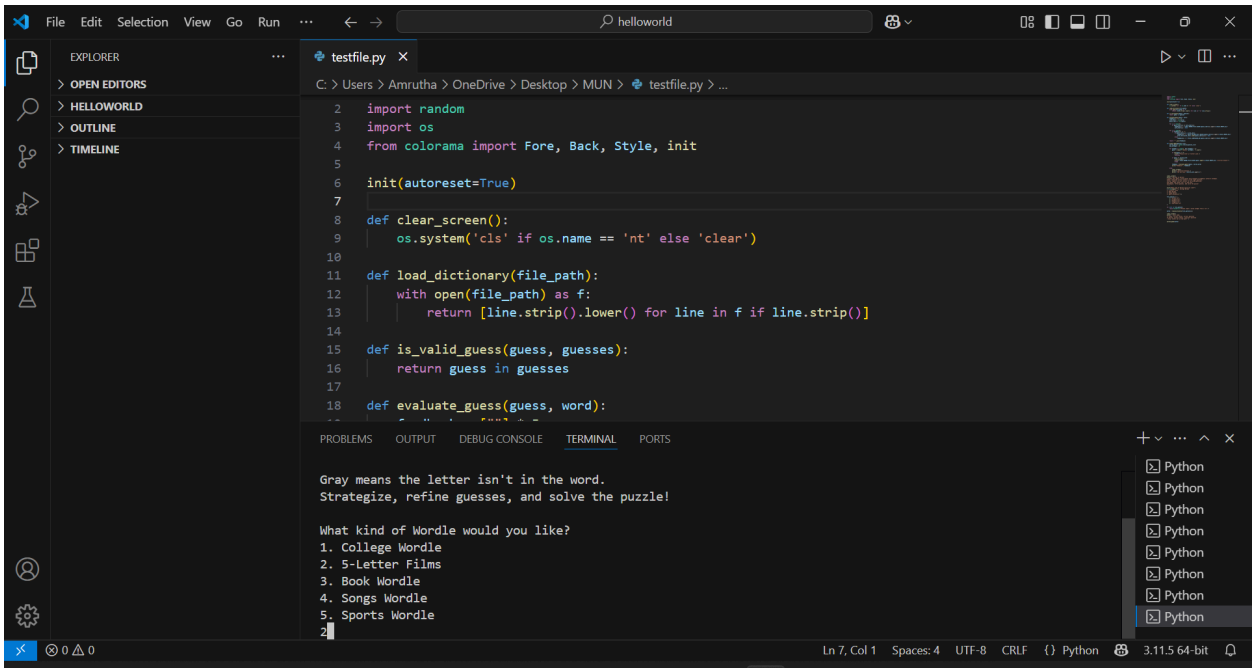


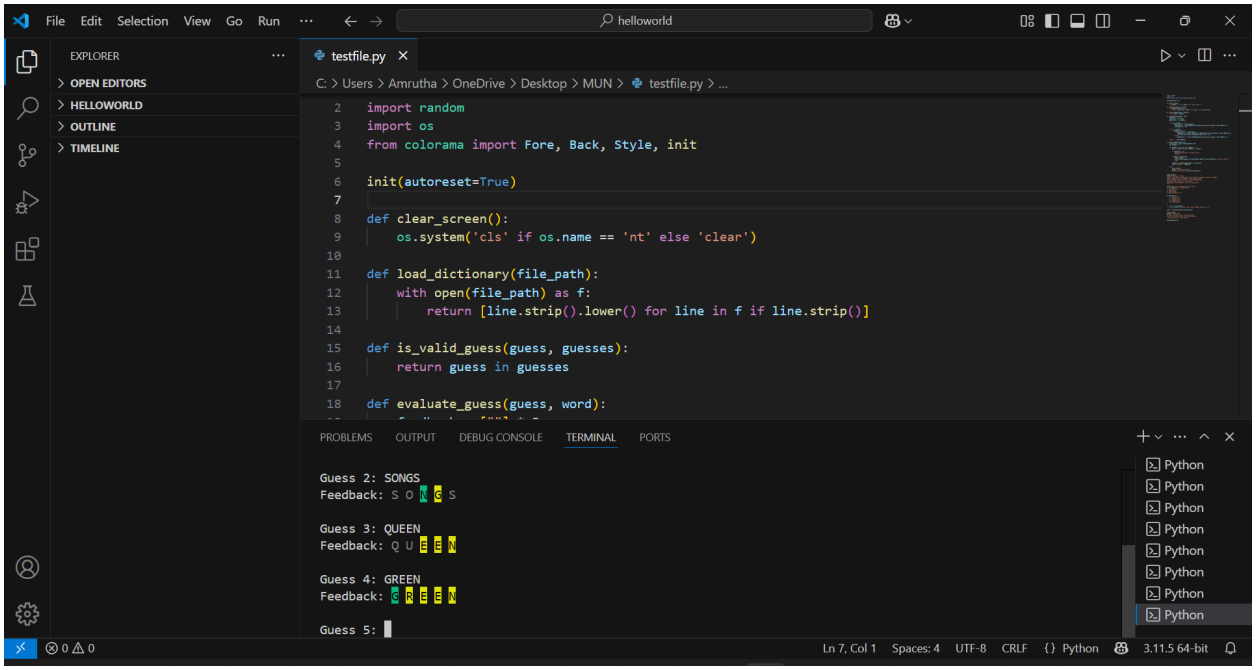


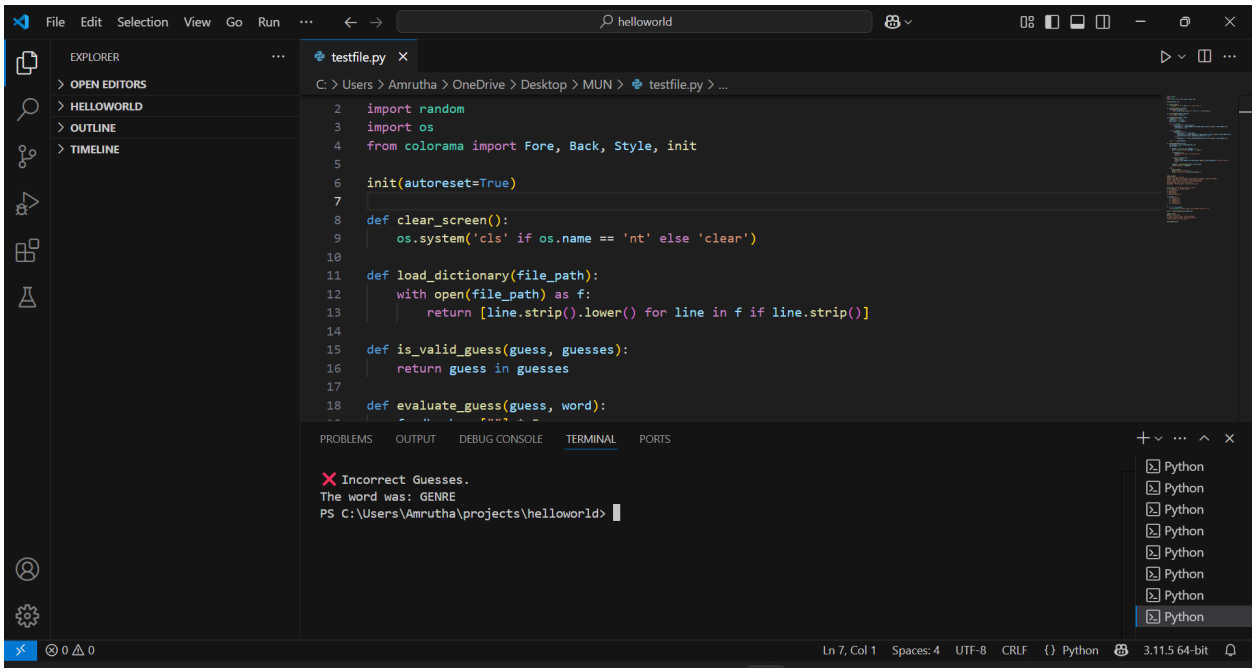
The program was able to effectively pick a random word from the text files, check if the guess was correct or not, provided color-coded feedback and cleared the program after it was over.

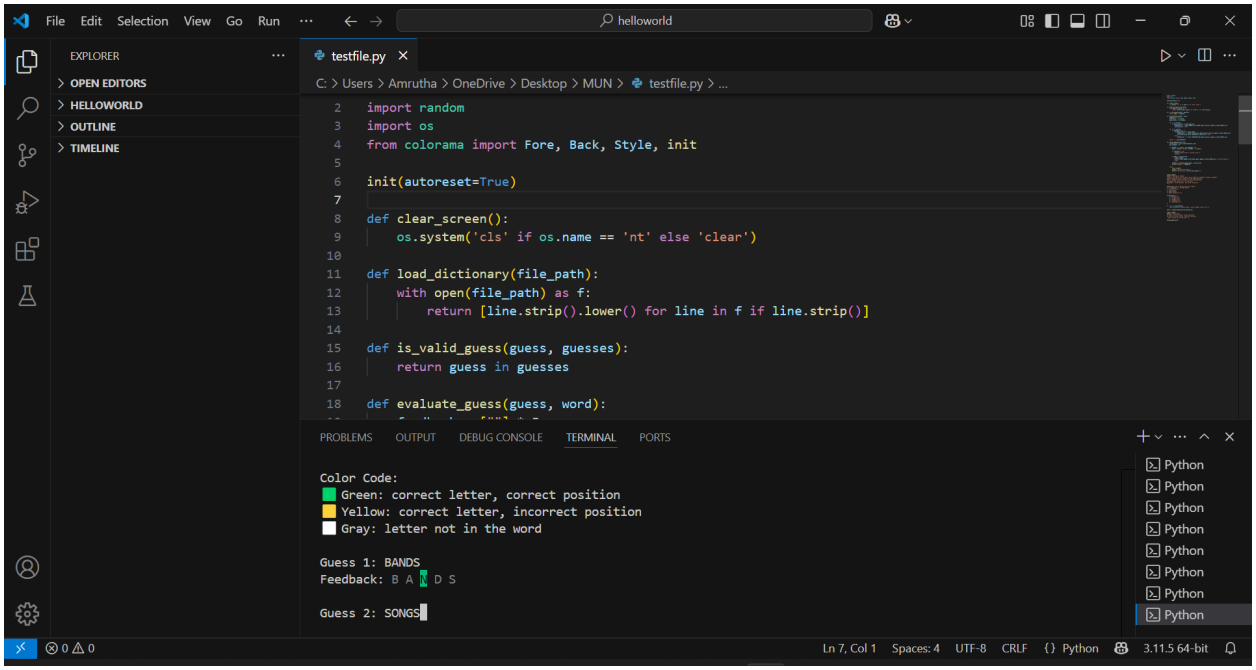
Snapshots

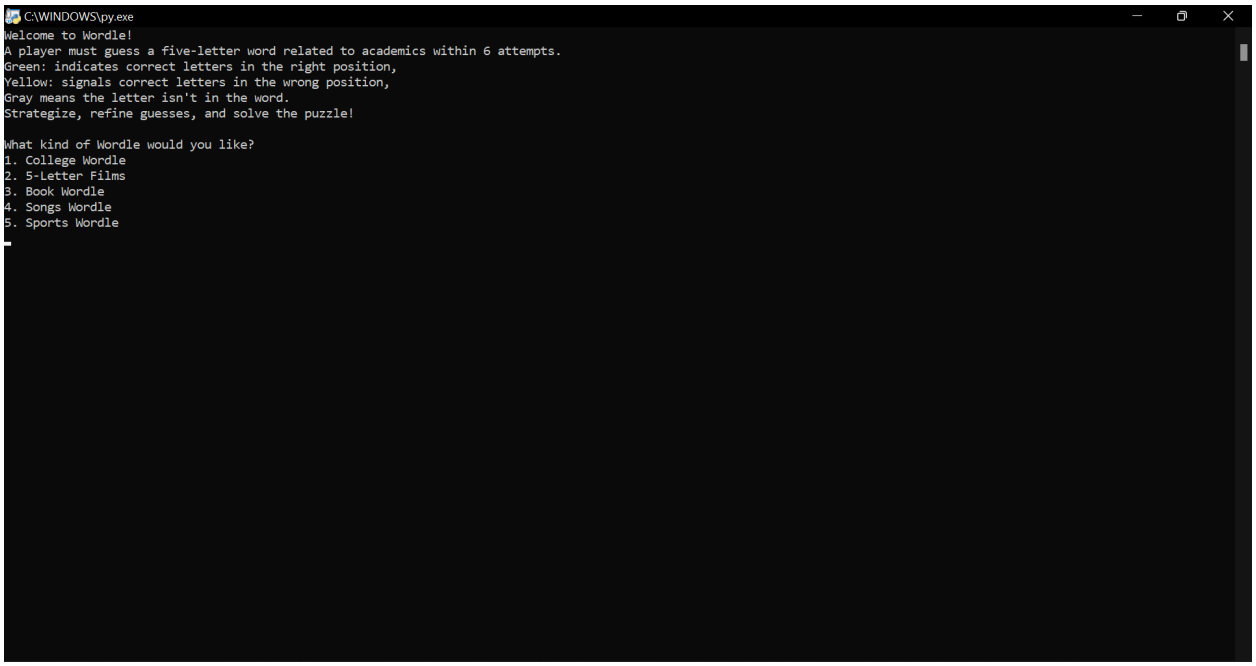

File Edit Selection View Go Run
helloworld
EXPLORER
testfile.py
OPEN EDITORS
HELLOWORLD
OUTLINE
TIMELINE
C: > Users > Amrutha > OneDrive > Desktop > MUN > testfile.py > ...
import random
import os
from colorama import Fore, Back, Style, init
init(autoreset=True)
def clear_screen():
os.system('cls' if os.name == 'nt' else 'clear')
def load_dictionary(file_path):
with open(file_path) as f:
return [line.strip().lower() for line in f if line.strip()]
def is_valid_guess(guess, guesses):
return guess in guesses
def evaluate_guess(guess, word):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Guess 1: BANDS
Feedback: B A N D S
Guess 2: SONGS
Feedback: S O N G S
Guess 3: QUEEN
Feedback: Q U E E N
Guess 4:
Python
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.11.5 64-bit


File Edit Selection View Go Run
helloworld
EXPLORER
testfile.py
OPEN EDITORS
HELLOWORLD
OUTLINE
TIMELINE
C: > Users > Amrutha > OneDrive > Desktop > MUN > testfile.py > ...
import random
import os
from colorama import Fore, Back, Style, init
init(autoreset=True)
def clear_screen():
os.system('cls' if os.name == 'nt' else 'clear')
def load_dictionary(file_path):
with open(file_path) as f:
return [line.strip().lower() for line in f if line.strip()]
def is_valid_guess(guess, guesses):
return guess in guesses
def evaluate_guess(guess, word):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Gray means the letter isn't in the word.
Strategize, refine guesses, and solve the puzzle!
What kind of Wordle would you like?
1. College Wordle
2. 5-Letter Films
3. Book Wordle
4. Songs Wordle
5. Sports Wordle
2
Python
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.11.5 64-bit


File Edit Selection View Go Run
helloworld
EXPLORER
OPEN EDITORS
HELLOWORLD
OUTLINE
TIMELINE
testfile.py
C: > Users > Amrutha > OneDrive > Desktop > MUN > testfile.py > ...
import random
import os
from colorama import Fore, Back, Style, init
init(autoreset=True)
def clear_screen():
os.system('cls' if os.name == 'nt' else 'clear')
def load_dictionary(file_path):
with open(file_path) as f:
return [line.strip().lower() for line in f if line.strip()]
def is_valid_guess(guess, guesses):
return guess in guesses
def evaluate_guess(guess, word):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Guess 2: SONGS
Feedback: S O N G S
Guess 3: QUEEN
Feedback: Q U E E N
Guess 4: GREEN
Feedback: G R E E N
Guess 5:
Python
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.11.5 64-bit


File Edit Selection View Go Run
helloworld
EXPLORER
OPEN EDITORS
HELLOWORLD
OUTLINE
TIMELINE
testfile.py
C: > Users > Amrutha > OneDrive > Desktop > MUN > testfile.py > ...
import random
import os
from colorama import Fore, Back, Style, init
init(autoreset=True)
def clear_screen():
os.system('cls' if os.name == 'nt' else 'clear')
def load_dictionary(file_path):
with open(file_path) as f:
return [line.strip().lower() for line in f if line.strip()]
def is_valid_guess(guess, guesses):
return guess in guesses
def evaluate_guess(guess, word):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
❌ Incorrect Guesses.
The word was: GENRE
PS C:\Users\Amrutha\projects\helloworld>
Python
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.11.5 64-bit


File Edit Selection View Go Run
helloworld
EXPLORER
OPEN EDITORS
HELLOWORLD
OUTLINE
TIMELINE
testfile.py
C: > Users > Amrutha > OneDrive > Desktop > MUN > testfile.py > ...
import random
import os
from colorama import Fore, Back, Style, init
init(autoreset=True)
def clear_screen():
os.system('cls' if os.name == 'nt' else 'clear')
def load_dictionary(file_path):
with open(file_path) as f:
return [line.strip().lower() for line in f if line.strip()]
def is_valid_guess(guess, guesses):
return guess in guesses
def evaluate_guess(guess, word):
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Color Code:
Green: correct letter, correct position
Yellow: correct letter, incorrect position
Gray: letter not in the word
Guess 1: BANDS
Feedback: B A N D S
Guess 2: SONGS
Python
Ln 7, Col 1 Spaces: 4 UTF-8 CRLF { } Python 3.11.5 64-bit


C:\WINDOWS\py.exe
Welcome to Wordle!
A player must guess a five-letter word related to academics within 6 attempts.
Green: indicates correct letters in the right position,
Yellow: signals correct letters in the wrong position,
Gray means the letter isn't in the word.
Strategize, refine guesses, and solve the puzzle!
What kind of Wordle would you like?
1. College Wordle
2. 5-Letter Films
3. Book Wordle
4. Songs Wordle
5. Sports Wordle

Conclusion

This project successfully demonstrates how Python can be used to build an engaging and interactive word-guessing game. By incorporating external libraries like Colorama for colored terminal output and utilizing fundamental programming concepts such as loops, conditionals, functions, and file handling, the program delivers a user-friendly experience similar to the classic Wordle game.

The addition of multiple word categories—such as college terms, films, books, songs, and sports—adds variety and allows players to test their vocabulary across different domains. Clear feedback, input validation, and limited attempts make the gameplay challenging yet fair.

References:

1. https://www.freecodecamp.org/news/building-a-wordle-game/
2. https://www.w3schools.com/python/python_ref_file.asp
3. https://pypi.org/project/colorama/
4. https://docs.python.org/3/library/os.html
5. https://www.w3schools.com/python/ref_random_choices.asp
6. Thareja, R. (2019). Python Programming: Using Problem Solving Approach. Oxford University Press, USA.