Subject: Computer Programming: Python

Faculty: Premanand S /SENSE / VIT-CC

FAT Question No: 5

Vaibhav and Manisha, during their Pongal holidays, decided to create a Python program to play the **HANGMAN Game**. Your task is to implement a similar game.

The HANGMAN game is a word-guessing game where players are presented with jumbled versions of words and must guess the correct word. The game will follow these rules:

Rules and Instructions:

1. At the start of the game, prompt the player to enter their **name**.
2. A word is randomly selected from a list of **5 complicated words** (you can hard-code these in the program).
3. The selected word will be **jumbled**, and the player will be shown this jumbled word.
4. The player has to guess the correct word.
    - If the player guesses correctly, they earn **1 point**.
    - If the player guesses incorrectly, they earn **0 points** for that round.
5. After each round, ask the player if they want to **continue** or **quit**:
    - If they choose to quit, display their total points and end the game.
    - If they choose to continue, proceed to the next word.

Example Interaction:

Player Name: Vaibhav
Word: "mlbeojd"
Player's Guess: "jumbled"
Result: "Correct! You earn 1 point."

Next Word: "nhegoma"
Player's Guess: "hangman"
Result: "Correct! You earn 1 point."

Continue or Quit? Quit

Final Score: 2

Solution:

Input:

1. **Player's Name:** The player enters their name at the start of the game.
2. **Jumbled Word:** The program selects a random word from a list and presents it in a jumbled format.
3. **Player's Guess:** The player guesses the correct word based on the jumbled word.

4. **Continue or Quit:** After each round, the player decides whether they want to continue the game or quit.

Output:

1. **Jumbled Word:** The jumbled version of the selected word is shown to the player.
2. **Result:** The program informs the player whether their guess is correct or incorrect.
3. **Score:** The program tracks and displays the player's score based on correct guesses.
4. **Final Score:** If the player chooses to quit, the program displays the total score accumulated throughout the game.

Processing:

1. **Word Selection:** A word is selected randomly from the predefined list of words.
2. **Word Jumble:** The selected word is shuffled to create a jumbled version.
3. **Guess Evaluation:** The player's guess is checked against the original word. If correct, the player earns 1 point; otherwise, no points are awarded.
4. **Score Calculation:** Points are accumulated based on correct guesses.
5. **Game Continuation/Termination:** After each round, the player is asked whether they want to continue or quit. If they quit, the game ends, and their score is displayed.

Alternate Approach:

This alternate solution introduces:

- **A predefined list of words**, where the program stores a dictionary with words and their meanings (to make the game more educational).
- **The game loop** continues by presenting a clue about the word's meaning after the word is jumbled.
- **The guess validation** is improved by giving feedback on the length of the guess, guiding the player through their guesses.

Algorithm:

1. **Start the game**:
    - Display a welcome message.
    - Prompt the player for their name.
    - Initialize the player's score to 0.
2. **Main Game Loop**:
    - Randomly select a word from the list of words.
    - Shuffle the selected word to generate a jumbled version.
    - Display the jumbled word to the player.
    - Get the player's guess and check if it's correct.
        - If the guess is correct, increment the score by 1 and display a success message.
        - If the guess is incorrect, reveal the correct word.
    - Ask the player whether they want to continue or quit.
        - If the player chooses to continue, repeat the process with the next word.
        - If the player chooses to quit, exit the loop and display the total score.

3. **End the game**:
    - Display the total score accumulated by the player.
    - Exit the program.

Code:

```
import random

# List of complicated words

words = ["complicated", "conundrum", "mysterious", "difficult", "puzzle"]

def jumble_word(word):

    """Returns a jumbled version of the word."""

    word_list = list(word)

    random.shuffle(word_list)

    return ''.join(word_list)

def hangman_game():

    # Step 1: Start the game

    print("Welcome to the Hangman Game!")

    player_name = input("Enter your name: ")

    total_score = 0

    game_over = False


    while not game_over:

        # Step 2: Randomly select a word

        word = random.choice(words)

        jumbled_word = jumble_word(word)


        # Display the jumbled word

        print(f"Jumbled word: {jumbled_word}")
```

```
# Step 3: Get player's guess
player_guess = input("Your guess: ").strip().lower()

# Validate the guess (ensure it's a word with letters only)
if not player_guess.isalpha():
    print("Invalid input! Please enter a valid word.")
    continue

# Step 4: Check if the player's guess is correct
if player_guess == word:
    print("Correct! You earn 1 point.")
    total_score += 1
else:
    print(f"Incorrect! The correct word was: {word}")

# Step 5: Ask if the player wants to continue or quit
continue_game = input("Continue or Quit? ").strip().lower()
if continue_game == 'quit':
    game_over = True
    print(f"Final Score: {total_score}")
elif continue_game != 'continue':
    print("Invalid input! Please type 'continue' or 'quit'.")
    continue
```

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

Output:

```
Welcome to the Hangman Game!
Enter your name: Premanand
Jumbled word: fifitudcl
Your guess: difficult
Correct! You earn 1 point.
Continue or Quit? Continue
Jumbled word: etapdcilomc
Your guess:
Invalid input! Please enter a valid word.
Jumbled word: zulezp
Your guess: hjs
Incorrect! The correct word was: puzzle
Continue or Quit? Continue
Jumbled word: tifculdfi
Your guess: jsdhasjk
Incorrect! The correct word was: difficult
Continue or Quit? Quit
Final Score: 1
```