Hangman: A C++ Implementation

Understanding the Code Structure and Functionality

Authors: Hena (65) & Rahul Kumar (85)

Date: 06/03/2025

Introduction

- Hangman is a word-guessing game where players try to save a character from being hanged by guessing letters.
- This document explains the C++ implementation with clear explanations and expected outputs.
- We will break down each part of the code and provide insights into its function.

Project Structure

- The project consists of three main files:
  1. `main.cpp` - Controls the game logic.
  2. `hangman_functions.h` - Declares functions used throughout the game.
  3. `hangman_functions.cpp` - Implements the declared functions.
- The program uses loops, conditional statements, and vectors to manage game logic effectively.

Header File (`hangman_functions.h`)

```
#include <iostream>
#include <vector>
#include <string>
using namespace std;

// Function declarations
void greet();
void display_misses(int misses);
void display_status(vector<char> incorrect, string answer);
void end_game(string answer, string codeword);
```

- This file serves as an interface between `main.cpp` and `hangman_functions.cpp`.
- Declares four main functions used in the game:
  1. `greet()`: Displays the game introduction.
  2. `display_misses(int misses)`: Shows the hangman figure.
  3. `display_status(vector<char> incorrect, string answer)`: Displays incorrect guesses and current progress.
  4. `end_game(string answer, string codeword)`: Determines the outcome of the game.

Function: greet()

```
void greet() {
    cout << "------------------\n";
    cout << "Hangman : The Game\n";
    cout << "------------------\n";
     cout << " Instructions : Save your friend by guessing the letters in the codeword.
\n";
}
```

- This function introduces the game to the player.

- Displays the game title and basic instructions.

- Example Output:

```
  ------------------
  Hangman : The Game
  ------------------
  Instructions: Save your friend from being hanged!
```

Main Program (`main.cpp`)

```
int main() {
    greet();
    string codeword = "codingwithcpp";
    string answer(codeword.length(), '_'); // Initialize answer with underscores
    int misses = 0;
    vector<char> incorrect;
    char letter;
```

- This is the starting point of the game.

- Initializes key variables:

  * `codeword`: The hidden word.

  * `answer`: A string of underscores representing unguessed letters.

  * `misses`: Tracks incorrect attempts.

- Example Output at this stage:

```
 _____________
```

Game Loop

```
while (answer != codeword && misses < 7) {
    display_misses(misses);
    display_status(incorrect, answer);
    cout << "\n\nPlease enter your guess: ";
    cin >> letter;
}
```

- This loop continues until the player guesses the word or makes 7 incorrect guesses.

- Calls `display_misses()` to show the hangman figure.

- Calls `display_status()` to display incorrect guesses and word progress.

- Example Output:

```
  Incorrect Guesses:

  _ _ _ _ _ _ _ _ _ _ _ _ _
  Please enter your guess:
```

Ending the Game

```
end_game(answer, codeword);
return 0;
```

- Calls `end_game()` to print the final result.
- Example Output (Win):

```
Hooray! You saved the person!
```

- Example Output (Lose):

```
Oh no! The man is hanged!
```