Programming Project / C++ – Hangman Manager and Player System –

Student 1: Stefania Sandu

Student 2: Mara Tudor

I. Task Description

Student 1 is responsible for managing the game:
- Adding/removing words to the word list
- Viewing the leaderboard and player history

Student 2 is responsible for interacting with the game:
- Playing the hangman game using the available words
- Saving the player's score to the leaderboard

II. Data Structures Used by the Team

The following data structures will be used:

- vector<string> wordlist - holds the in-memory word list
- vector<pair<string, int>> leaderboard - stores (name, score) pairs
- Files are used to persist data between the two applications

III. File Structure

The following files will be used:

words.txt
List of available words for gameplay (1 per line)
<word 1>
<word 2>
<word 3>
…

leaderboard.txt
Each line contains <player_name>,<score>

IV. Interacting with Executables

Application 1 will offer the following options:

./app_1.exe add_word <word>

To add a word to the list of available Hangman words

./app_1.exe delete_word <word>
To delete a word from the list

./app_1.exe view_words
To view all available words

./app_1.exe view_leaderboard
To view all scores saved by previous players

./app_1.exe view_history <player_name>
To view all past scores for a specific player

Application 2 will offer the following options:

./app_2.exe play

Launches the Hangman game loop. A random word is selected from words.txt.
The player guesses letters until the word is solved or 5 incorrect guesses are made.
Final score is saved to leaderboard.txt

Both apps communicate through shared files:

- words.txt: is written by App1 and read by App2. It's purpose is to supply playable words.
- leaderboard.txt is written by App2 and read by App1. It's purpose is to store and display scores.

This forms a bidirectional flow:

- App1 prepares the game environment
- App2 plays the game and sends back results