Quiz Game – Technical Description

This document updates the original description of the command-line-based quiz game inspired by "Who Wants to Be a Millionaire?" to include the newly implemented features and command-line arguments.

Project Structure

- include/
  - Question.h
  - QuizGame.h
- src/
  - QuizGame.cpp
- main1.cpp (Gameplay)
- main2.cpp (Management)
- Makefile
- questions.txt

New Functionalities

- 50/50 Lifeline: Removes two incorrect answers for a given question.

- Session Saving: Tracks player progress question-by-question in a temporary file named `<name>_session.txt`.

- Score Saving: Final score is saved to a leaderboard.txt file when a session ends.

- Output File: Stores player name, score, and date in output.txt after completion.

- View Leaderboard: Displays all players and their scores, sorted descending.

- View History: Shows the history of a particular player from leaderboard.txt.

- Add Question: Allows the addition of new questions with four answers and the index of the correct one.

- Full command-line argument validation: Invalid or unknown arguments trigger helpful error messages.

Data Structures Used

-struct Question- Represents a quiz question and its associated answers.

-struct LeaderboardEntry- Stores a player's name and score for leaderboard tracking.

- std::vector<Question> questions- Holds all loaded questions for gameplay.

- std::vector<LeaderboardEntry> leaderboard- Temporarily stores all leaderboard entries for sorting and display.

Command-Line Arguments

--menu (main1)
Displays available commands and usage examples.

--play (main1)
Starts/resumes the game session.

--q=<number> (main1)
Specifies which question to load (indexed from 1).

--name=<player_name> (main1)
The name of the player. Used for progress and score tracking.

--use5050=<0|1> (main1)
Whether to use the 50/50 lifeline (1) or not (0).

--answer=<0-4> (main1)
The player's answer. 0 means no answer is being given, just showing the question.

--leaderboard (main2)
Displays all scores saved in leaderboard.txt.

--history --name=<player_name> (main2)
Shows all entries of a specific player.

add <question> <option1> <option2> <option3> <option4> <correctIndex> (main2)
Adds a new question to the quiz.

Task Distribution

Student 1: Mara Herman

-Developed gameplay logic in `main1.cpp`

-Implemented 50/50 lifeline logic

-Added session saving and question flow

-Handled output file creation for score and date

-Designed error handling for invalid answers

Student 2: Armagan Tashan

-Developed management logic in `main2.cpp`

-Implemented leaderboard and history features (sorted descending)

-Added question addition functionality

-Improved argument validation and error messaging

-Modularized the project structure using header/source separation