|  |  |
| --- | --- |
| **Course code and name:** | **B37VB - Praxis Programming - 2024-2025** |
| **Type of assessment:** | **Individual** |
| **Coursework Title:** | **Game project – MindMatch** |
| **Student Name:** | **Anas Ramy Ibrahim Ahmed Ahmed Gomaa** |
| **Student ID Number:** | **H00481128** |

|  |
| --- |
| **Declaration of authorship. By signing this form:**   * **I declare** that the work I have submitted for individual assessment OR the work I have contributed to a group assessment, is entirely my own. I have NOT taken the ideas, writings or inventions of another person and used these as if they were my own. My submission or my contribution to a group submission is expressed in my own words. Any uses made within this work of the ideas, writings or inventions of others, or of any existing sources of information (books, journals, websites, etc.) are properly acknowledged and listed in the references and/or acknowledgements section. * I confirm that I have read, understood and followed the University’s Regulations on plagiarism as published on the [University’s website](https://www.hw.ac.uk/uk/students/studies/examinations/plagiarism.htm), and that I am aware of the penalties that I will face should I not adhere to the University Regulations. * I confirm that I have read, understood and avoided the different types of plagiarism explained in the University guidance on [Academic Integrity and Plagiarism](https://heriotwatt.sharepoint.com/sites/skillshub/SitePages/Academic-Integrity-and-Plagiarism.aspx)   **Student Signature** *(type your name): Anas Gomaa*  **Date**: *13/04/2025* |

Copy this page and insert it into your coursework file in front of your title page.

For group assessment each group member must sign a separate form and all forms must be included with the group submission.

**Your work will not be marked if a signed copy of this form is not included with your submission.**

Heriot-Watt University Dubai

B37VB - Praxis Programming - 2024-2025

|  |  |
| --- | --- |
| **Student’s name** | **Anas Gomaa** |
| **Student ID** | **H00481128** |
| **Professor’s name** | **Dr Girish Balasubramanian** |
| **Projects name** | **Game Project Dubai** |
| **Game’s name** | **MindMatch** |

**Development and Implementation of an Interactive Quiz Game in C**

**Abstract :**

The creation of an engaging and instructive interactive quiz game in the C programming language is presented in this lab report. Three game modes—single, 1v1, and multiplayer—as well as three categories—IQ, Math, and English—are supported. To maintain fairness, the game uses randomized question ordering and dynamic question loading based on user selection. A built-in timer restricts individual and one-on-one sessions to six minutes, and questions are randomized to maintain fairness. In IQ Test mode, a scoring system assesses user performance and allocates results to pre-established IQ ranges. The system's dependability in handling numerous players, precisely calculating points, and efficiently selecting winners based on accuracy and reaction time has been validated by extensive testing.

**Introduction :**

This project seeks to challenge the norm in a digital age where games frequently put entertainment value ahead of intellectual merit. This program's main objective is to create an entertaining and instructive quiz game in C that mimics aptitude and intelligence tests, which are frequently used instruments in psychological evaluation. This trivia game aims to enhance the player's thinking abilities and mental flexibility through meaningful interaction, in contrast to many contemporary games that can be addictive or provide little cognitive stimulation.

In addition to offering a variety of gameplay options, such as single-player, 1v1, and multiplayer modes, the program supports three distinct categories: IQ, Math, and English. Because players can compete and challenge friends, this not only offers a personalized experience but also promotes social interaction.

**Methodology:**

All of the code for the quiz game was implemented in a single source file (Game2.c), and it was created using standard C programming techniques. In addition to three categories (IQ Test, Math Quiz, and English Quiz) and three gameplay modes (solo, 1v1, and multiplayer), the design places a high priority on modularity, fairness, and user engagement.

Quiz content is stored using a structured data model. The question text, four possible answers, and the correct answer index are all stored in a question struct. The loadQuestions function loads questions based on the chosen category after they are hardcoded directly into the program.

Three separate functions—playSolo, play1v1, and playMultiplayer—are used to control gameplay. These functions handle time tracking, question flow, input collection, and scoring. Players use terminal prompts (scanf) to interact with the game. Arrays are used to collect and store names, responses, and mode choices. To gracefully handle inaccurate or out-of-range entries, input validation is used.  
Questions are shuffled using srand(time(0)) to implement randomization. Time\_t and difftime are used to enforce timing, and in solo and 1v1 modes, each player has a 6-minute limit.

While the IQ Test mode uses a tiered classification system to map scores to predefined IQ ranges, scoring for the Math and English categories is simple. Results with matching intelligence labels are shown by the displayIQScore function. In competitive modes, time efficiency and score are combined to determine the winner.

**Results :**

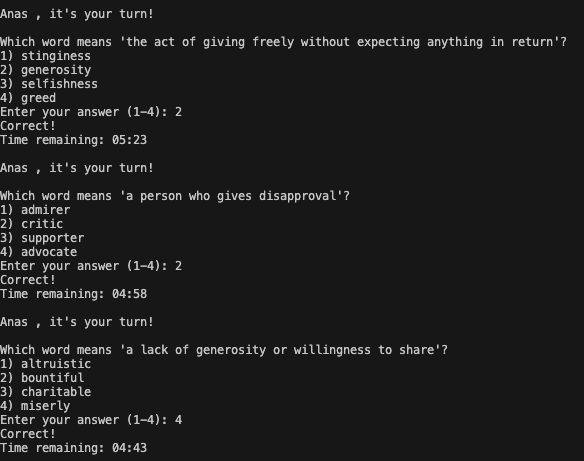
In order to assess functionality, user interaction, timing accuracy, and scoring logic, the program underwent extensive testing across all categories and gameplay modes.

Testing verified that the application loads questions dynamically according to the user's chosen category (English Quiz, Math Quiz, or IQ Test). For use in gameplay, the loadQuestions function appropriately selects the pertinent subset of hardcoded questions and saves them in memory.

A screenshot of a video game

AI-generated content may be incorrect.

Using srand(time(0)), randomization is carried out consistently, resulting in a distinct question order for every session. This improved replayability and fairness by guaranteeing that users encountered distinct and unpredictable gameplay throughout each run.

A screenshot of a computer

AI-generated content may be incorrect.

Both solo and one-on-one modes were used to test timing mechanisms. The six-minute time limit per player was precisely enforced by the difftime function. The game automatically terminated the session and showed the results if a player went over the limit.





All categories' scores were confirmed. DisplayIQScore was used to accurately map test results to IQ ranges, and users were provided with qualitative feedback (such as "Exceptional Genius"). Scores on math and English quizzes were determined by dividing the total number of correct answers by the number of incorrect answers.





In both 1v1 and multiplayer modes, the winner determination logic worked as planned. Players received clear feedback and were ranked according to their time and score.

A screen shot of a video game

AI-generated content may be incorrect.

Users thought the game was fun, responsive, easy to use, and had a good educational element overall.

**Discussion:**

Strong structural design and useful instructional elements are evident in this quiz game. With features like loadQuestions and playSolo, its modular design enhances code readability and streamlines maintenance in the future. Every session is unique due to the questions' randomization and the dynamic category selection. While the IQ Test mode adds value by mapping scores to standardized IQ categories, the support for Solo, 1v1, and Multiplayer modes improves user engagement.

Notwithstanding these advantages, certain drawbacks are apparent. The program's 100 hardcoded questions limit its scalability and content flexibility. The source code must be changed in order to update or expand the question bank. Furthermore, there is limited error handling, especially for unexpected or non-numeric inputs made during player setup, even though there is some input validation.

Future versions could enhance the program by enabling users to import custom questions from external files, which would make the content more dynamic and manageable. A customized challenge for varying skill levels could be offered by introducing adaptive difficulty. Stability would be improved by improved error handling, particularly in multiplayer settings. Lastly, a graphical user interface (GUI) might improve the game's visual appeal and accessibility for a wider range of players.

**Conclusion :**

This project effectively illustrates the planning and execution of an interactive quiz game created in C that blends entertaining gameplay elements with educational goals. In addition to offering three customizable gameplay modes—solo, 1v1, and multiplayer—the game supports three different quiz categories: IQ Test, Math Quiz, and English Quiz. These characteristics make the game appropriate for a broad spectrum of users by facilitating both social competition and individual learning.

Essential features like randomized question order, dynamic question loading, an integrated timing mechanism, and an easy-to-use scoring system were all executed flawlessly. By relating player scores to standardized IQ ranges, the IQ Test category provides more depth and cognitive feedback that goes beyond conventional quiz formats.

Simple terminal prompts are used for user interaction, and feedback is always clear and responsive. The program's dependability in setting time limits, generating precise scores, and selecting winners based on speed and performance was validated through testing.

Future updates could increase the game's impact even though the current version depends on a fixed, hardcoded question bank. A graphical user interface, adaptive difficulty levels, enhanced error handling, and external question loading are just a few of the improvements that would greatly increase its usability, flexibility, and appeal and turn it into an effective teaching tool.

**References:**

How2Become Ltd. (2014). IQ and Aptitude Tests. 2nd ed. How2Become Ltd. ISBN:

<https://www.how2become.com>

cplusplus.com . (n.d.). C Library Reference. [online] Available at:

<https://cplusplus.com>

TutorialsPoint. (n.d.). C Standard Library Functions. Available at: <https://www.tutorialspoint.com> .