Harry Potter Trivia

Ayush Jha

(590026090)

December 2 , 2025

Abstract:

In this project, I created a Harry Potter–themed quiz game using the C programming language. The program asks the user multiple-choice questions related to Harry Potter and keeps track of their score based on the answers. I used the following C programming concepts and commands: standard input and output functions like printf() and scanf(), if-else statements, loop, Variables, etc At the end, the program displays the total score and a message like “You’re a true wizard!” or “Better luck next time, Muggle!” based on the user’s performance.

Contents

1 PROBLEM DEFINITION

1.1 Overview

The Harry Potter Quiz Game is a simple and interactive C program designed to test the user's knowledge of the Harry Potter universe. The program presents a series of multiple-choice questions, takes user input, and evaluates the correctness of each answer. It keeps track of the user's score throughout the quiz and displays the final result at the end, along with a personalized message based on performance.

1.2 Objectives

1. To create an interactive multiple-choice quiz based on the Harry Potter series using the C programming language.
2. To practice and demonstrate core C concepts such as variables, loops, conditional statements, and input/output functions.
3. To implement a scoring system that evaluates the user's answers and displays performance-based feedback.
4. To use file handling to store the user's name and score in a persistent scoreboard file.
5. To build a simple, user-friendly console program that responds to user input and provides immediate results.
6. To develop a small-scale application that strengthens understanding of logic building and structured programming in C.

2 SYSTEM DESIGN

2.1 System Architecture

1. User Interaction: Handles all input and output, including displaying questions, taking answers, and showing the final score.

2. Quiz Logic: Runs the main loop of seven questions, checks answers, and updates the score based on correctness.

3. File Handling: Saves the user's name and score into *scoreboard.txt* using basic C file operations.

2.2 Flowchart

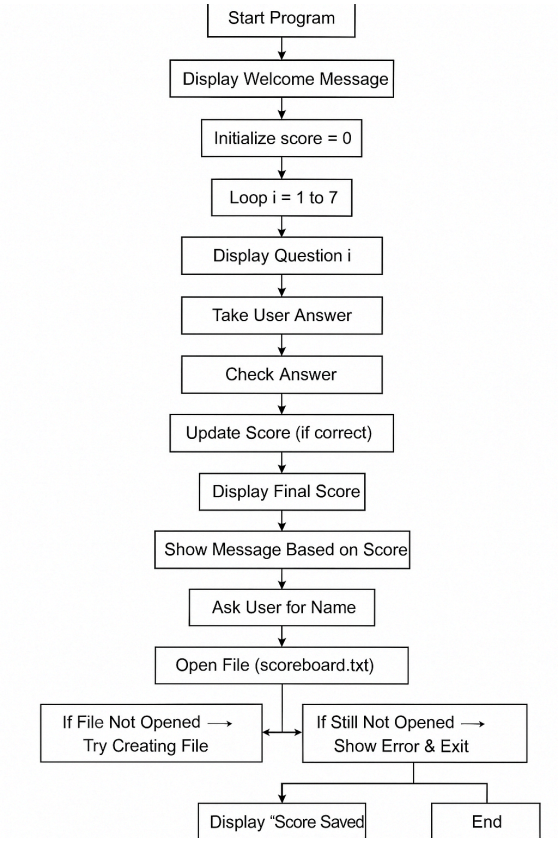

Figure 1: System Flowchart

2.3 Data Structures

```
int score;
int answer;
int i;
char name[50];
FILE *fp;
```

3 IMPLEMENTATION DETAILS

3.1 Key Features

1. Multiple-Choice Quiz System
2. Score Tracking
3. Performance-Based Feedback
4. User Name Input
5. File Handling for Score Saving
6. Simple and User-Friendly Console Interface

3.2 Code Snippets

3.2.1 Displaying the Welcome Message:

```
printf("Welcome to the Harry Potter Trivia Quiz!\n");

printf("Let's see how well you know the Wizarding World!\n\n");
```

3.2.2 Question Loop

```
for(i = 1; i <= 7; i++) {
    printf("Question %d...\n", i);
    printf("Enter your choice: ");
    scanf("%d", &answer);
    if(answer == correctOption)
        score++;
}
```

3.2.3 Final Score and Feedback

```
printf("Your final score is: %d out of 7\n", score);
if(score == 7)
    printf("You're a true wizard!\n");
else if(score >= 4)
    printf("Not bad! You're on your way to Hogwarts.\n");
else
    printf("Better luck next time, Muggle!\n");
```

3.2.4 Saving the Score to File

```
printf("\nEnter your name to save your score: ");
scanf(" %[^\n]", name);
fp = fopen("scoreboard.txt", "a");
if(fp == NULL)
   fp = fopen("scoreboard.txt", "w");
fprintf(fp, "%s - %d/7\n", name, score);
fclose(fp);
```

4 TESTING & RESULTS

4.1 Test Cases

Test Case 1: Correct Score Saving

Input: The user answers 3/7 questions correctly and enters the name “ayush”.

Expected Output:

Score should be saved in *scoreboard.txt* as:

Result: Passed.

Test Case 2: Perfect Score

Input: User answers all 7 questions correctly and enters the name "arman".

Result: Passed.

Test Case 3: Mid-Level Score

Input: The user answers 5/7 questions correctly and enters the name "rishi".

Result: Passed.

5 CONCLUSION & FUTURE WORK

5.1 Conclusion

The Harry Potter Quiz Game successfully meets its goal of providing an interactive and engaging quiz experience using the C programming language. The program demonstrates fundamental programming concepts such as loops, conditional statements, user input, and file handling. It accurately evaluates the user's answers, provides meaningful feedback, and stores quiz scores for future reference. Overall, the project showcases a clear understanding of structured programming and serves as a strong foundation for building more advanced applications.

5.2 Future Work

- Add more questions or randomize questions for increased replay value.
- Implement difficulty levels such as easy, medium, and hard.
- Introduce a graphical user interface (GUI) for a more interactive experience.
- Include sound effects or animations to enhance engagement.
- Store scores in a more structured format such as CSV or a database.
- Add a feature to view previous scores directly within the program.

6 REFERENCES

1. Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language. Prentice Hall.
2. Lafore, R. (1997). Object-Oriented Programming in Ct +. Sams Publishing.
3. Online resources and documentation