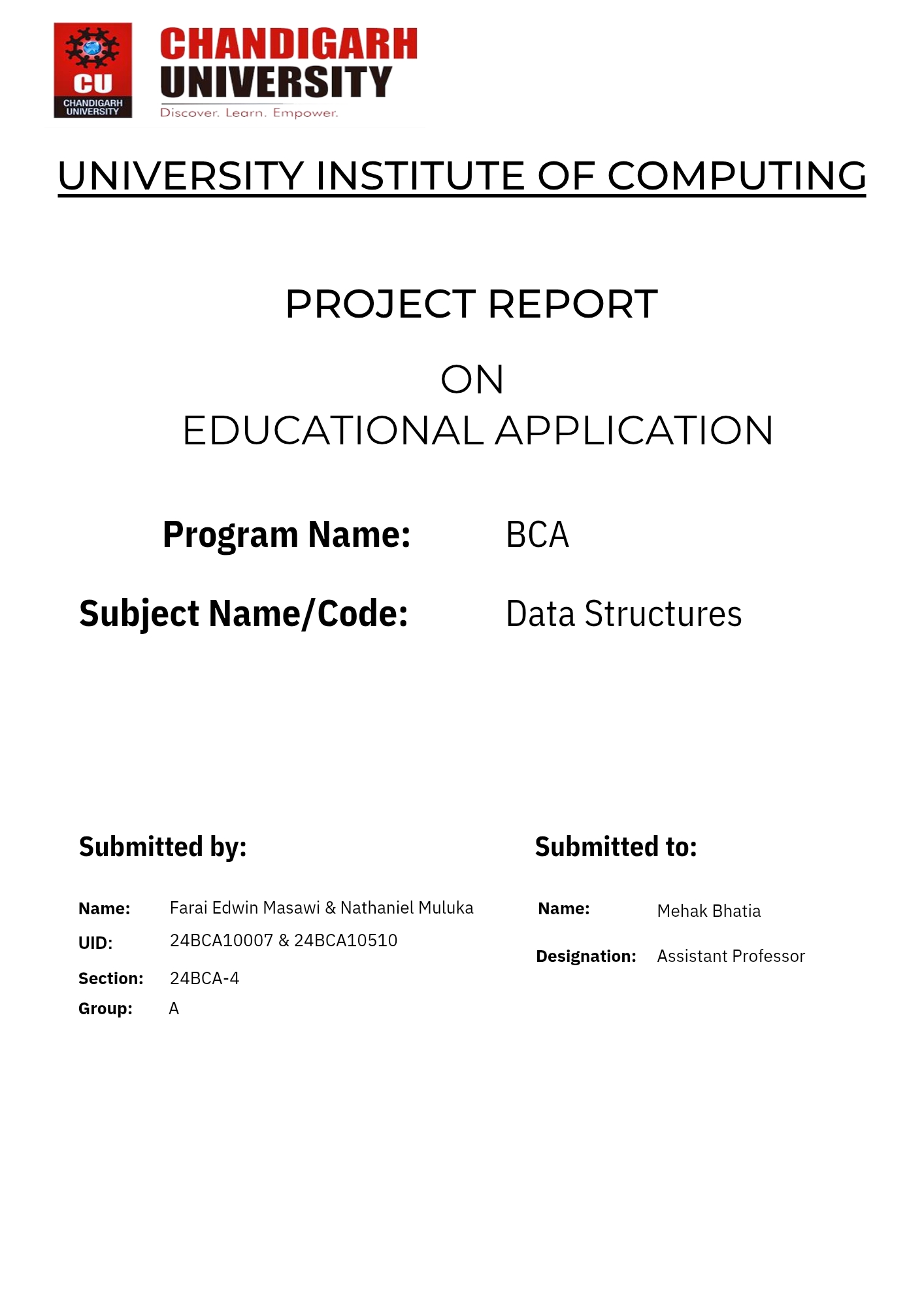
****

**ABSTRACT**

### Introduction:

This C-based quiz application is designed to help students practice and understand the **Queue data structure** through a gamified learning system. It demonstrates both **educational content delivery** and **data structure implementation**, specifically focusing on **queue applications**. The program tests the user's understanding with 10 multiple-choice questions, explanations for each, and a grading system based on performance.

### Technique:

This project was implemented in **C language**, using:

1. **Structures (`struct`)** to define a `QuizQuestion` object with attributes for the question, options, correct answer, and explanation.
2. **Control Structures & Loops** for quiz flow, answer validation, and tracking remaining lives.
3. **Queue Concept Integration** as the educational theme of the quiz—questions test real-world applications of different queue types (Simple Queue, Circular Queue, Deque, Priority Queue).
4. **Grading Logic** to evaluate performance and provide motivational feedback based on score percentages.

The code employed primarily uses the Structure (struct) data structure in C, and internally, it also leverages arrays. More succinctly, struct + arrays (1D and 2D) form the main data structures in play here.

### System Configuration:

* **OS:** Android (Termux) / Linux (online compiler)
* **Compiler**: GCC
* **Memory**: Lightweight, under 1 MB usage
* **Editor**: GitHub Mobile + Online C Compilers

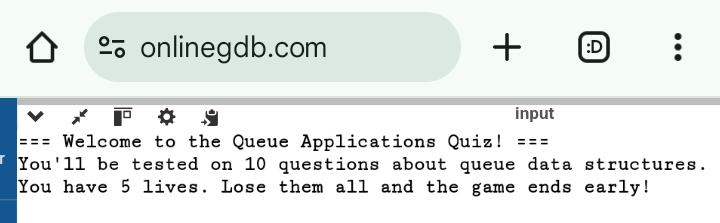
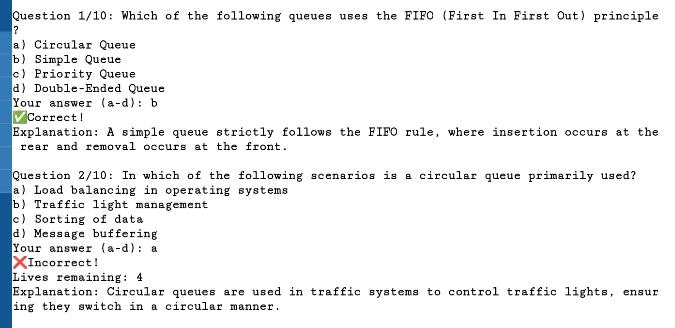
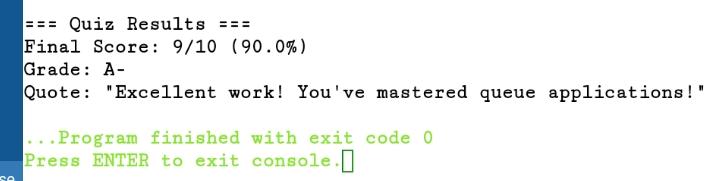
**SUMMARY**

### Input:

The user is guided through the following:

* Instructions and welcome screen.
* Answering each of the **10 questions**, with options (a-d).
* A total of **5 lives** are given. A wrong answer deducts a life.
* After each question, an **explanation** is displayed to reinforce learning.

### Sample Input:



**Process**

***Algorithm****:*

1. **Start**.
2. **Initialize Constants:**

* `NUM\_QUESTIONS = 10`
* `MAX\_OPTIONS = 4`
* `LIVES = 5`

1. **Define Structure `QuizQuestion`:**

* Fields: `question`, `options`, `correct\_answer`, `explanation`

1. **Create an Array `quiz[NUM\_QUESTIONS]`**\*\*

* Each element stores one `QuizQuestion` with question, options, correct answer, and explanation.

1. **Display Welcome Message**
2. **Initialize:**

* `lives = LIVES`
* `score = 0`

1. **For each question (I = 0 to NUM\_QUESTIONS – 1) AND while `lives > 0`:**
2. Display current question number and question text
3. Display the 4 options labeled a–d
4. Prompt the user to enter their answer (a, b, c, or d)
5. Convert the answer to lowercase
6. **If answer is correct:**

* Display “Correct”
* Increment `score` by 1

1. **Else:**

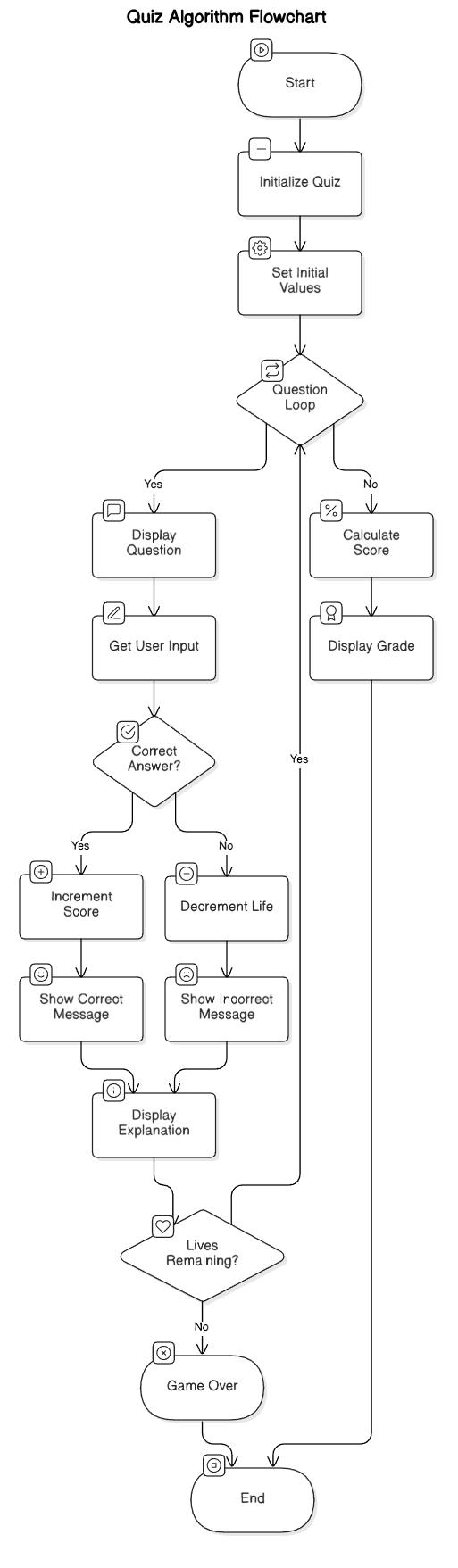
* Display “Incorrect”
* Decrement `lives` by 1
* Display remaining lives

1. Show the explanation for the question.
2. **If `lives == 0`:**

* Display “Game Over” message

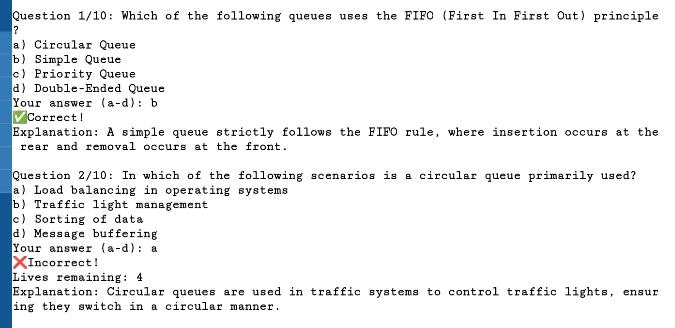
1. **Call `displayGrade(score)` function:**
   1. Calculate percentage: `(score / NUM\_QUESTIONS) \* 100`
   2. Display score and percentage
   3. Display appropriate grade and quote based on percentage
2. **End**.

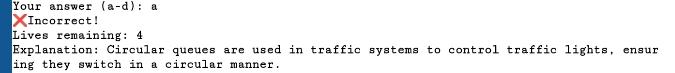
***Algorithm Flowchart****:*

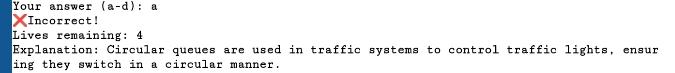
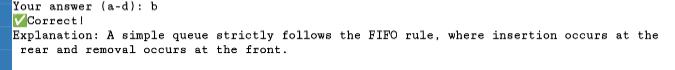


**OUTPUT**

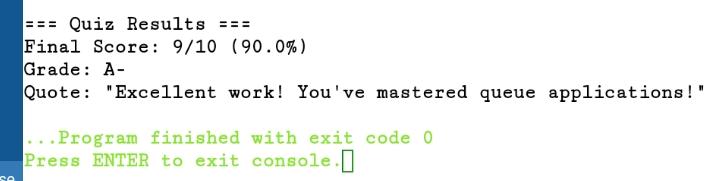
* **Correct/Incorrect response** shown immediately.

****

* **Lives tracking** on wrong answers.
* **Explanations** after each question.

****

* **Final Score and Grade** with feedback.



**UNIQUE ASPECTS**

* **Educational Use of Queues:** The project itself teaches queue concepts, going beyond just using data structures.
* **Feedback-Oriented Learning:** Each wrong answer still provides the correct explanation.
* **Gamified Engagement:** Using lives and grades to simulate a "game-over" system keeps users engaged.
* **Expandable Template**: The code structure allows easy addition of more topics (e.g., Stacks, Trees, Sorting).

**FUTURE SCOPE**

* Convert this CLI quiz into a **web-based app** using AJAX or API methods.
* Allow **custom quiz topics**, **question banks**, and **user login** to track progress.
* Integrate with **GitHub Pages** for online access to educational tools on Data Structures.