

**UNIVERSITY OF ASIA PASIFIC**

Department of Computer Science & Engineering

**Course Title:** Computer Graphics Lab

**Course code:** CSE 426

**Project:** Catch the Eggs

**Submitted To**

Jayonto Dutta Plabon

Lecturer,

CSE, UAP

**Submitted By**

Labanya Saha (21201059)

Jannatun Saumoon (21201066)

Paria Chowdhury (21201067)

**Introduction & Game Concept:**

**Game Overview:**

“Catch the Eggs”

It is a 2D computer graphics-based arcade game built using OpenGL and GLUT. The player controls a basket placed at the bottom of the screen to catch falling eggs dropped by moving hens. The goal is to collect as many normal and golden eggs as possible while avoiding poop within a limited time.

**Gameplay Rules:**

* Normal Egg = +5 points
* Golden Egg = +10 points
* Poop = -10 points
* Missing any good egg = -3 points
* The timer starts at 60 seconds. The game ends when the timer reaches 0.

**Graphics Concepts Applied:**

* Use of 2D transformations for hen movement and leg animation.
* GLUT primitive shapes (circle, ellipse, rectangle, triangle) for game elements.
* Color gradients and simple animation for grass and clouds.
* Double buffering for smooth animation using GLUT\_DOUBLE.
* Event handling for keyboard and mouse input.

**Implementation Details:**

**Tools and Libraries:**

* Language: C
* Libraries: OpenGL (GL, GLU, GLUT,winmm,freeglut)
* Platform: Windows
* Additional API: mmsystem.h (for built-in Windows sounds)
* IDE: Code::Blocks

**Code Structure:**

* The project Catch the Eggs is developed using **C language** with **OpenGL (GLUT)** for 2D graphics rendering.
* The program is divided into separate functional sections to maintain clarity and teamwork coordination.
* At the beginning, all required **libraries, constants, and global variables** are declared.
* The **drawing section** contains functions to display background, hens, eggs, poops, basket, and decorative elements.
* The **logic section** manages gameplay functions such as egg dropping, scoring, movement, and timing.
* The **sound and interface section** controls menus, buttons, and audio effects.
* The **main function** initializes the OpenGL window, sets callbacks, and runs the main game loop.

**Main Modules**

The project is divided into three main modules

1. **Graphics & Environment Module**

This module is responsible for all the visual design and animations of the game.  
It focuses on creating an attractive and realistic game environment.

**Key Functional Parts:**

* **Background Design:**  
  Designed the sky, grass, trees, and barn to give a complete farm-like appearance.  
  Multiple layers were used to create depth and perspective.
* **Object Drawing:**  
  Separate functions were created to draw hens, eggs, poops, basket, and the mascot hen using OpenGL primitives.  
  Each object was given specific colors and proportions for better clarity.
* **Animation Handling:**  
  Hens were animated to move slightly and look alive.  
  Eggs were given smooth falling motion using timer functions.
* **Display Integration:**  
  All objects were rendered in the main display function, ensuring a continuous and fluid frame update.

**Purpose:**  
To make the game visually appealing and to give players a smooth animated experience.

1. **Interface & Sound Module**

This module handles the **user interface and sound system** of the game.  
It ensures that players can easily interact with the game and enjoy background music and effects.

**Key Functional Parts:**

* **Menu and Navigation:**  
  Designed multiple interface screens such as Play, Settings, Instructions, and Exit.  
  The menu allows smooth navigation using keyboard or mouse input.
* **Game Over and Restart Screens:**  
  A separate page is displayed when the game ends, showing the final score and options to restart or quit.
* **Audio Management:**  
  Used the Windows PlaySound() function to play background music and sound effects.  
  Different sounds were used for events like catching an egg, missing one, or game over.
* **Visual Text and Layout:**  
  Used readable fonts, colors, and text alignment to make the interface attractive and clear.

**Purpose:**  
To enhance user engagement through well-designed screens and appropriate sound feedback.

1. **Logic & Gameplay Module**

This is the **core module** that controls all the game rules, movements, and scoring logic.  
It ensures that the game behaves correctly according to player actions.

**Key Functional Parts:**

* **Egg Spawning System:**  
  Eggs are generated randomly from hens at different intervals and fall toward the basket.  
  Random speed and type (normal, golden, or poop) make the game more dynamic.
* **Basket Movement Control:**  
  The basket moves horizontally using keyboard input.  
  Boundary checking prevents the basket from going off-screen.
* **Collision Detection:**  
  The system continuously checks if an egg touches the basket.  
  Catching increases score; missing reduces life or plays a “miss” sound.
* **Scoring and Timer:**  
  Scores are updated for every catch, and a countdown timer controls the total play time.  
  When time becomes zero, the game automatically ends.
* **State Management:**  
  Maintains transitions between Menu, Game, Pause, and Game Over states.

**Purpose:**  
To manage all gameplay activities and maintain proper coordination between movement, timing, and scoring.

**Algorithms Implemented**

Several simple but effective algorithms have been implemented to make the gameplay smooth and realistic:

* **Egg Spawning & Falling Algorithm:**  
  Eggs are randomly generated from hens at different intervals. Each egg falls vertically at a fixed speed until it is either caught by the basket or missed on the ground.
* **Collision Detection Algorithm:**  
  The game continuously checks whether a falling egg overlaps with the basket area. If the collision occurs, the score increases; otherwise, missing an egg may reduce life or points.
* **Basket Movement Algorithm:**  
  The basket can be moved horizontally using keyboard keys. The movement is restricted within the screen boundary so that the basket cannot go out of view.
* **Scoring and Timer Algorithm:**  
  Every time an egg is caught, the score is updated. A timer continuously counts down during the game, and when the time reaches zero, the game automatically ends and shows the final result.
* **State Management Algorithm:**  
  The program maintains different game states such as *menu, play, instruction,* and *game over*. Depending on user input or game conditions, it switches between these states smoothly.
* **Sound Trigger Algorithm:**  
  Whenever a particular event occurs (like catching or missing an egg), specific system sounds are played using built-in Windows audio functions to make the game more engaging.

**Game Features**

**1.Main Menu**

* The game starts with a colorful farm-style menu.
* Buttons for Play, Settings, Instructions, Score, and Exit make it easy to navigate.
* When the mouse moves over a button, it reacts with a light hover effect.

**2.Instructions Screen**

* Shows gameplay rules and control keys.
* Displays egg types with their point values:
  + Normal Egg → +5 points
  + Golden Egg → +10 points
  + Poop → −10 points
  + Missed Good Egg → −3 points
* Explains keyboard and mouse controls.

**3.Score Page**

* Displays the top five high scores.
* Updates automatically when a new score is higher than the old ones.
* Has a clear “Press Esc to return to menu” note for easy navigation.

**4.Gameplay Screen**

* **Farm-themed environment** with sky, clouds, grass, barn, and fence.
* Multiple **animated hens** moving side to side, dropping eggs randomly.
* Player controls a **basket** at the bottom to catch eggs.
* **Egg Types:** Normal, Golden, and Poop, each affecting score differently.
* **Smooth movement** using keyboard (A/D or arrow keys) or mouse.
* **Live score** and **countdown timer** displayed at the top.
* **Pause/Resume** feature with Spacebar.
* Built-in **sound effects** for catching, missing, or game over events.

**5.Game Logic & Scoring System**

* +5 for Normal Egg, +10 for Golden Egg.
* −10 for catching Poop.
* −3 for missing a Normal/Golden Egg.
* Game duration: **60 seconds**.
* Automatic transition to **Game Over** when time runs out.

**6.Game Over Screen**

* After the timer hits zero, a dark “Game Over” window appears.
* It shows the player’s final score and asks for their name.
* Three buttons appear: Retry, Quit, and Menu.
* High scores update instantly.

**7.Sound Effects**

* Uses Windows system sounds — no extra files needed.
* Catching an egg plays a light “ding.”
* Missing an egg or catching poop plays a warning tone.
* A deeper sound marks the end of the game.

**8.Animations and Graphics**

* Hens walk and move their legs as if they’re alive.
* Clouds drift slowly across the sky.
* Grass waves gently to give a sense of motion.
* Everything is drawn with OpenGL shapes like circles, ellipses, and rectangles.

**9.Controls**

* **Keyboard:**
  + A / Left Arrow → Move left
  + D / Right Arrow → Move right
  + Spacebar → Pause or resume
  + Esc → Go back to menu
* **Mouse:**
  + Move the basket by moving the mouse left or right.
  + Click to select buttons in menus.

**10.Extra Touches**

* Random egg drops keep every game different.
* Real-time updates make the motion smooth.
* You can restart instantly without closing the window.
* The whole game feels light and cheerful, with bright colors and simple movement.

**Individual Contributions:**

Each Member’s Contribution-

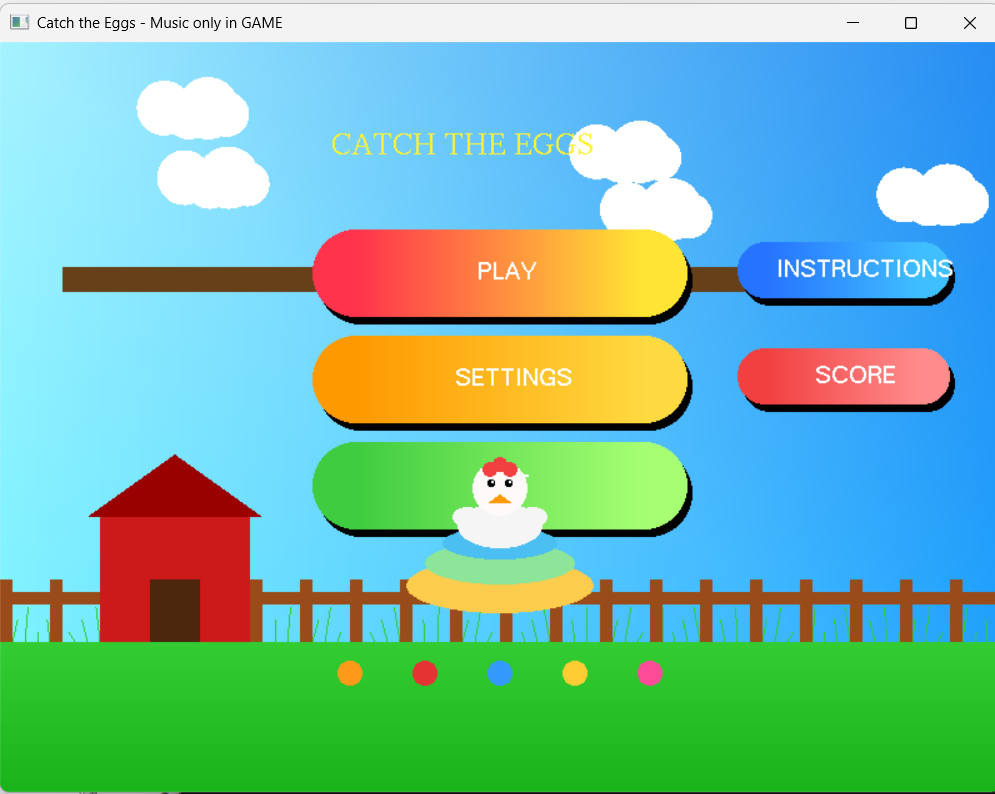
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Members** | **Module** | **Task Performed** | **Key Features** | **Focus Area** |
| Jannatun Saumoon (66) | Graphics & Environment Developer | - Designed background and environment elements (sky, grass, barn, trees)  - Drew hens, eggs, poops, basket, and mascot hen  - Created smooth animations using OpenGL timer functions  - Integrated all visuals into the display function | - Background design and object rendering  - Animation control for hen and eggs  - Visual consistency during gameplay | Visual design and environment creation |
| Paria  Chowdhury (67) | Interface & Sound Designer | - Designed Main Menu, Settings, Instructions, and Game Over screens  - Added navigation and button features  - Implemented background music and event-based sound effects  - Styled fonts and colors for a clean user layout | - Menu and interface system  - Sound integration using PlaySound()  - UI text and layout management | User interaction and sound feedback |
| Labanya Saha (59) | Logic & Gameplay Programmer | - Developed core game logic and control flow  - Implemented egg spawning and falling algorithm  - Programmed basket movement with keyboard control  - Wrote collision detection and scoring logic  - Managed timer, life count, and game states | - Game logic and rule implementation  - Collision & scoring system  - State management and timer control | Gameplay logic and performance |
| All Members | Team Collaboration | - Discussed game structure and visual flow  - Tested each module separately and then integrated them  - Fixed minor bugs during final testing | - Coordinated design, logic, and sound for smooth integration | Team coordination and testing |

**Challenges Faced & Solutions:**

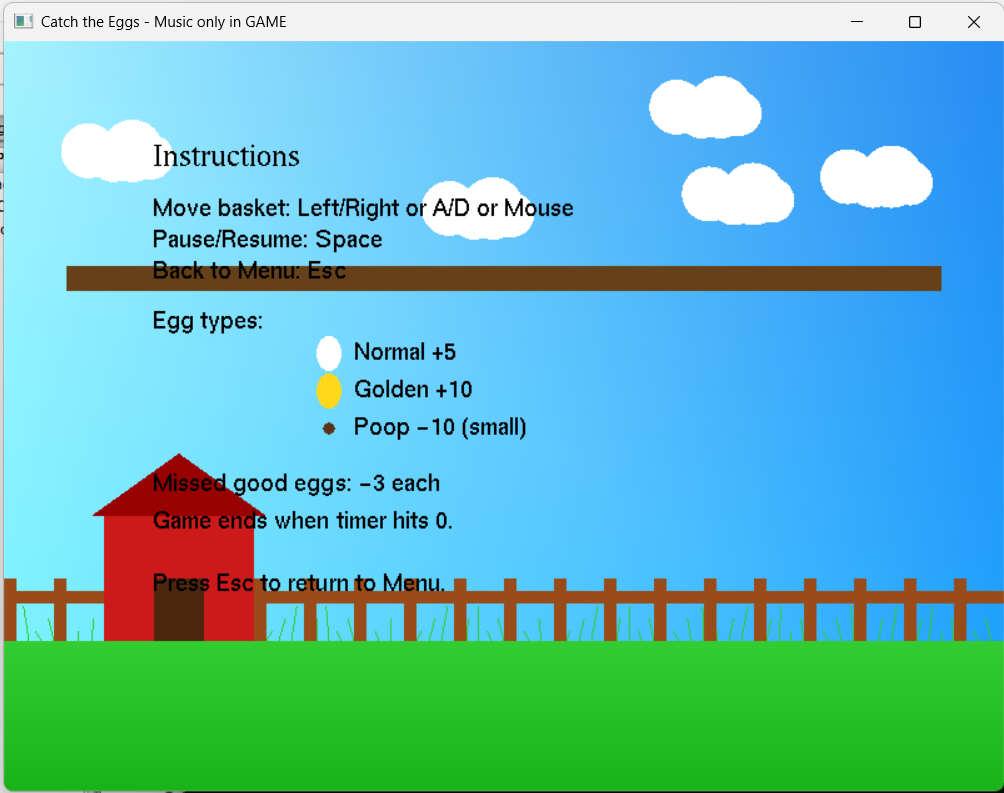
|  |  |
| --- | --- |
| **Challenges** | **Solutions** |
| **Hen size & movement logic** –  Adjusting hen size and fixing egg-laying was tricky | Used OpenGL transformations and step-by-step debugging to scale the hen correctly and implement smooth movement and egg drop events. |
| **Grass animation & background** –  Moving grass and adding music was complicated. | We implemented grass movement with timers and incremental updates; for background sound, we replaced music with Windows notification sound. |
| **Sound integration** –  Adding extra sound effects caused issues. | We managed sound using Windows PlaySound() function directly in the code |
| **Basket representation** –  Changing basket design in OpenGL wasn’t straightforward | We created the basket using OpenGL shapes as a simple alternative. |

**Final Output**

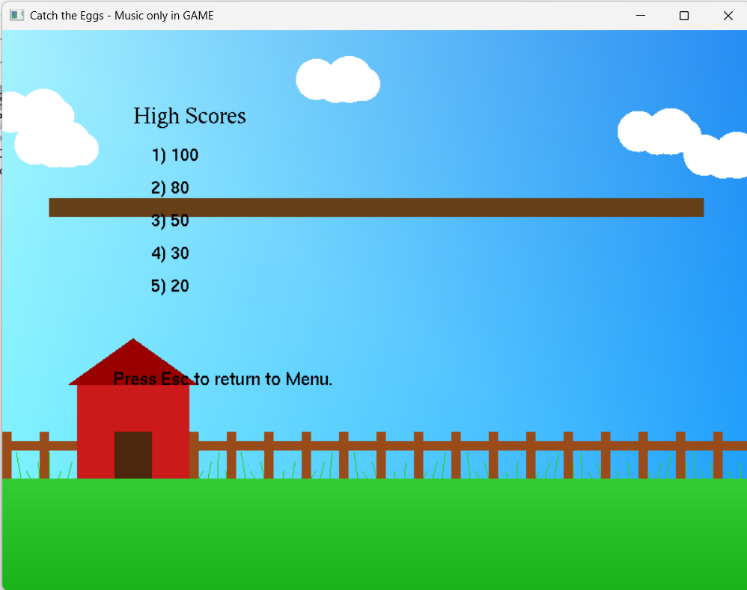
**Interface**



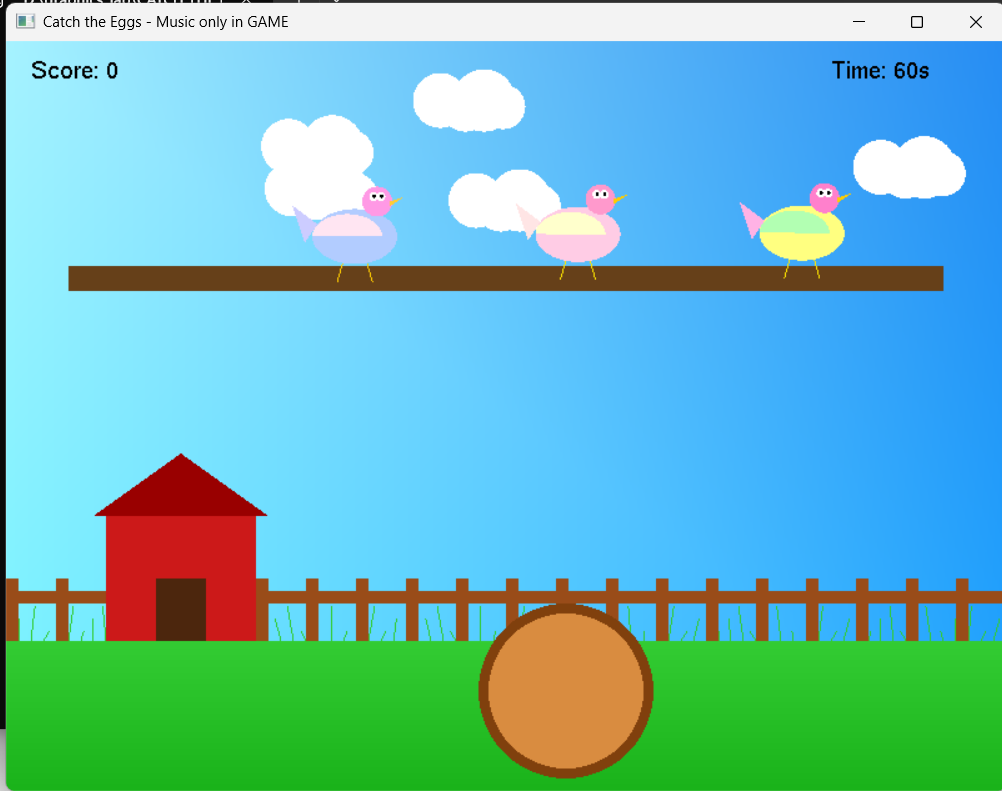
**Fig:-Menu**



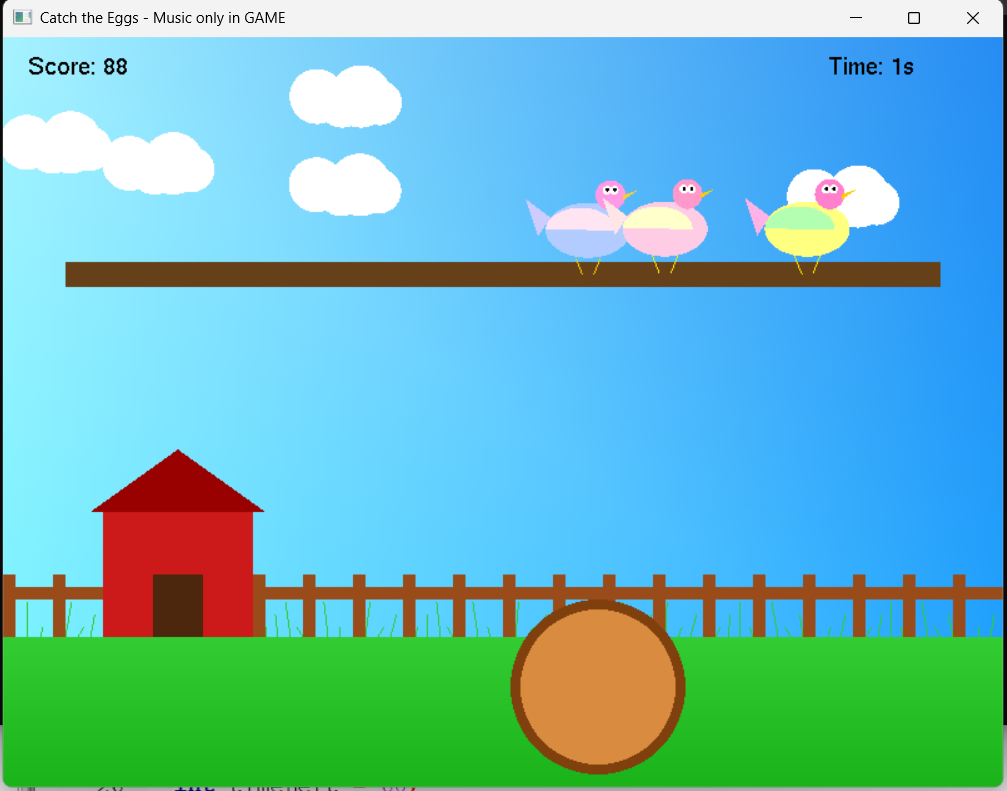
**Fig:-Instruction**



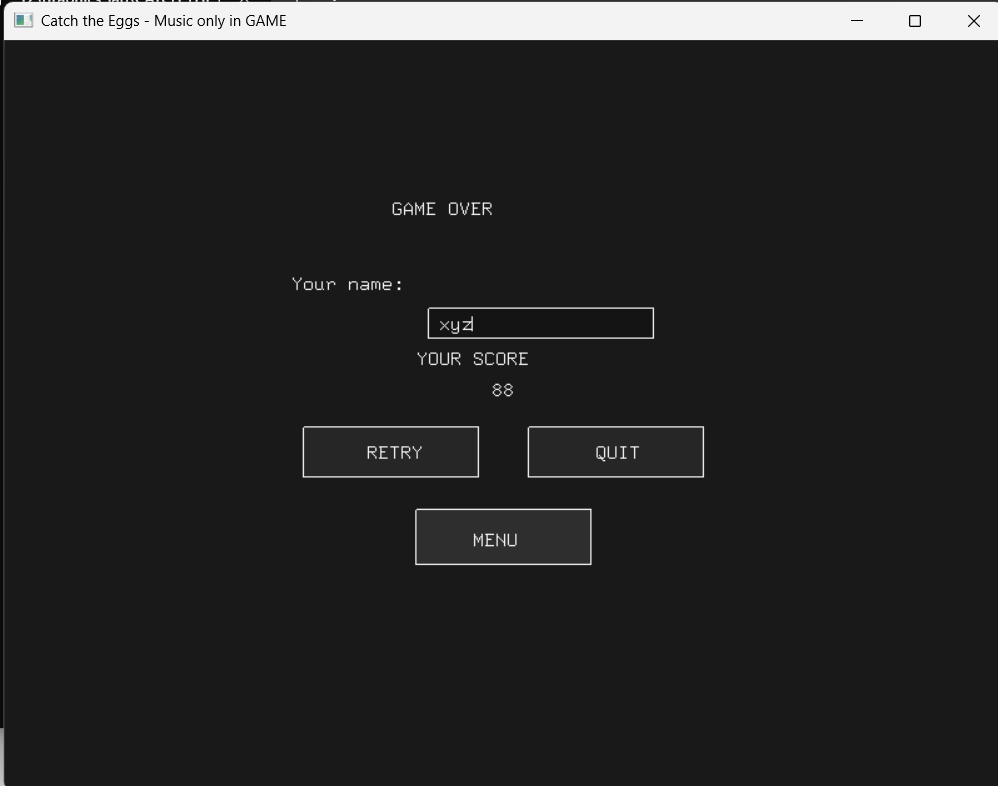
**Fig:-score**



**Fig:-Game start**



**Fig:-Game finish**



**Fig:-Game Over Interface**

**Conclusion**

* + - The project successfully delivers a complete 2D farm-themed catching game built using **C and OpenGL.**
    - It helped us understand how **OpenGL and GLUT** can be used to create real-time visuals and event-driven programs.
    - The game successfully integrates **animation, user input, sound, and scoring** into one complete system (Animated hens, moving clouds and grass, and multiple egg types make the game visually engaging.)
    - Each team member worked on a separate module — graphics, sound/UI, and gameplay logic — making teamwork effective and organized.
    - The final output runs smoothly with clear visuals, responsive controls with both **keyboard and mouse,** and basic sound effects.
    - Through this project, we learned practical concepts of **designing, debugging, and integrating** different parts of a graphical program.

Overall, this project demonstrates the successful combination of **graphics, logic, and sound integration** of creating a simple, fun, and fully functional in a single C-based OpenGL program.

**Future Improvements**

* Add continuous background music and volume control settings.
* Introduce multiple levels or difficulty modes to make the game more challenging.
* Design a more detailed **Settings Page** — where players can adjust basket speed, timer duration, and toggle sound effects.
* Improve the **UI design** with better button layouts, hover animations, and theme options.
* Store player names and high scores permanently using a file or small database.
* Add more animated characters and dynamic backgrounds for variety. And implement advanced visual effects like shadows or lighting using modern OpenGL.