FAST NUCES , Karachi Campus

DSA LAB PROJECT PROPOSAL

Maryam Saleem 24K-0779

Umaima Manzoor 24K-0501

Aiman Misbah 24K-0842

Project Report:

Mini Arcade Game Collection Project Title: Mini Arcade Game Collection – Hangman, Flappy Bird, Maze & Main Menu

Group Members:

Maryam Saleem – 24K-0779

Umaima Manzoor- 24K-0501

Aiman Misbah – 24K-0842

Submission Date: 22nd November 2025

1. Executive Summary

Overview: This project presents three arcade-style games which uses assembly language and simple visuals to present object-oriented programming concepts and user interaction. The collection includes Maze, Flappy Bird, Hangman, and a Main Menu system that integrates all the games. Each game confirms output graphics, game logic, and input handling.

Key Findings:

- Hangman portrays conditional logic, scoring, and string manipulation.
- Flappy Bird emphasizes game mechanics, crash detection, and real-time input handling.
- Grid navigation, key-based movement, and mini-map visualization were built in the Maze game.
- All games have been linked and navigable via Main Menu.

2. Introduction

Background: Reusability, modular design, and structured code are made possible by object-oriented programming. Mini-games provides in engaging way to apply OOP concepts like event-driven design, modularity, and encapsulation. Using assembly language and basic graphics for console-based user interaction, this project deals with these ideas.

Project Objectives:

- Build a series of minigames that highlight OOP concepts
- Delivering an interactive menu for navigating and selecting games for the players to play
- Provides players with usability and clear visual feedback.

3. Project Description Scope:

Scope:
The project is built on three games and a central menu system Each game includes full input handling and output rendering features. Advanced graphical rendering, online leaderboards, and network-based multiplayer features are not covered in the project.

Technical Overview:

Development Environment: Visual Studio community

Language used is x86 Assembly (Irvine32 library)

Tools: text positioning, console color manipulation, and input/output library Irvine32.inc

4. Methodology

Approach:

- The project was completed in several sessions:
- The division was done based on separate games such as Hangman, Flappy bird and Maze. Each game was built separately to keep the logic clear and simple and to avoid conflicts.
- Each feature of the games was tested by each member before moving to next step.
- The integration of game was done in a way so that each game is manipulated via main menu

Roles and Responsibilities:

- Umaima Manzoor: Main Menu system, Flappy Bird implementation and overall integration.
- Maryam Saleem: Hangman game and string input handling.
- Aiman Misbah: Maze game design and implementation

5. Project Implementation

Design and structure:
Each game is built separately for clarity and smooth functioning. Through main menu the player can navigate to any of the three games. The output is displayed on console screen for the graphical representation of the game . The control flow is kept simple to avoid any register conflicts or ambiguity.

Functionalities Developed:

Hangman : Word selection , string input handling, life counts and updating the display

Flappy bird: The bird in flappy bird moves up and down , shifitng of obstacles and collision prevention

Maze : Movements of player , checking of boundaries and exit logic

Challenges Faced:

- Managing registers when switching between code blocks
- Displaying the output on console screen as assembly provides very limited formatting
- Handling continuous movements of player in maze and bird in flappy bird
- Careful string and memory handling for letters, attempts and lives

6. Results

Project Outcomes:

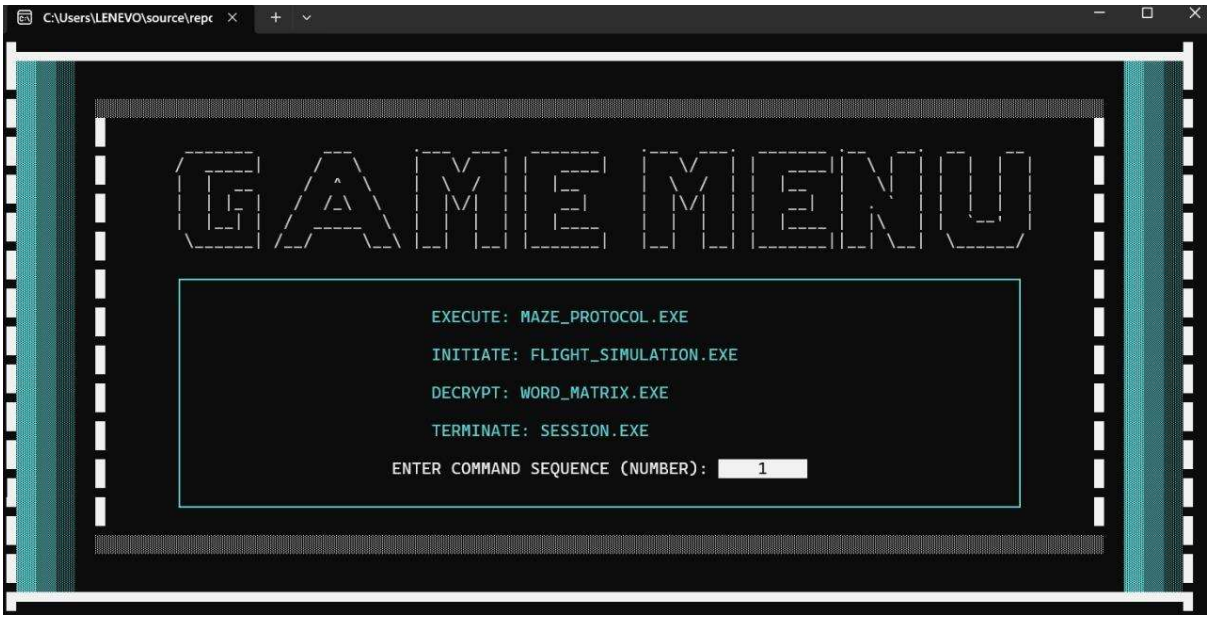
All four games can run smoothly and efficiently with transitioning loading screen and no runtime crashes reported
Hangman: valid and invalid guesses are handled perfectly with deduction in lives
Flappy bird: UP and DOWN movements of the bird and collision (obstacle) handling
Maze: check for boundaries and perfect movements of the players
Main menu: Smooth transition and navigation to each game

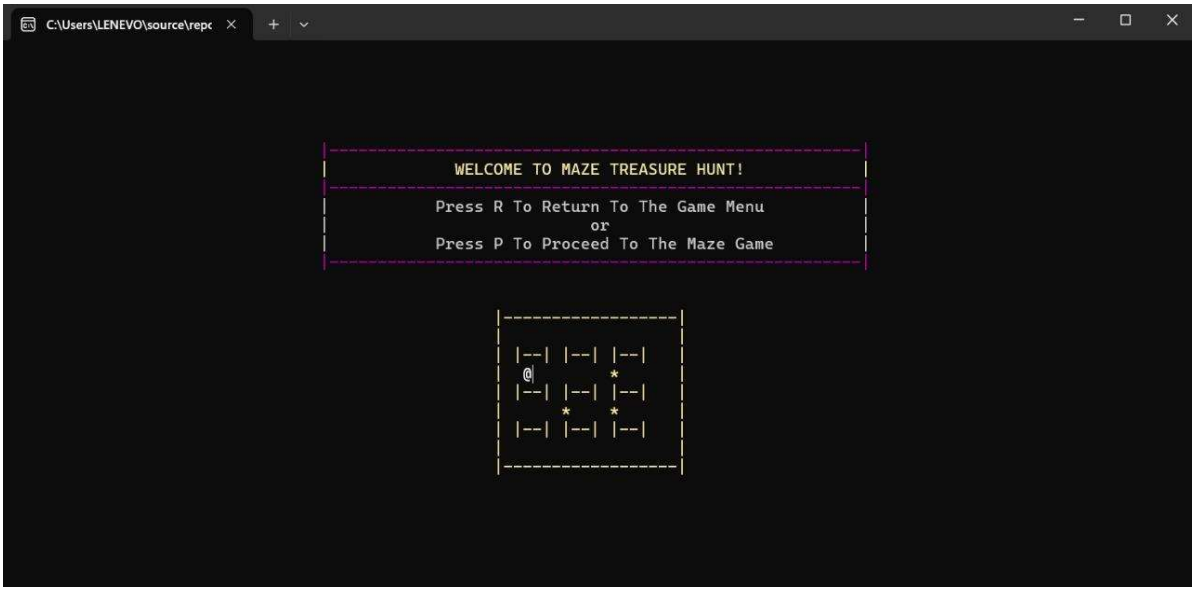
Screenshots and Illustrations :

MAIN MENU

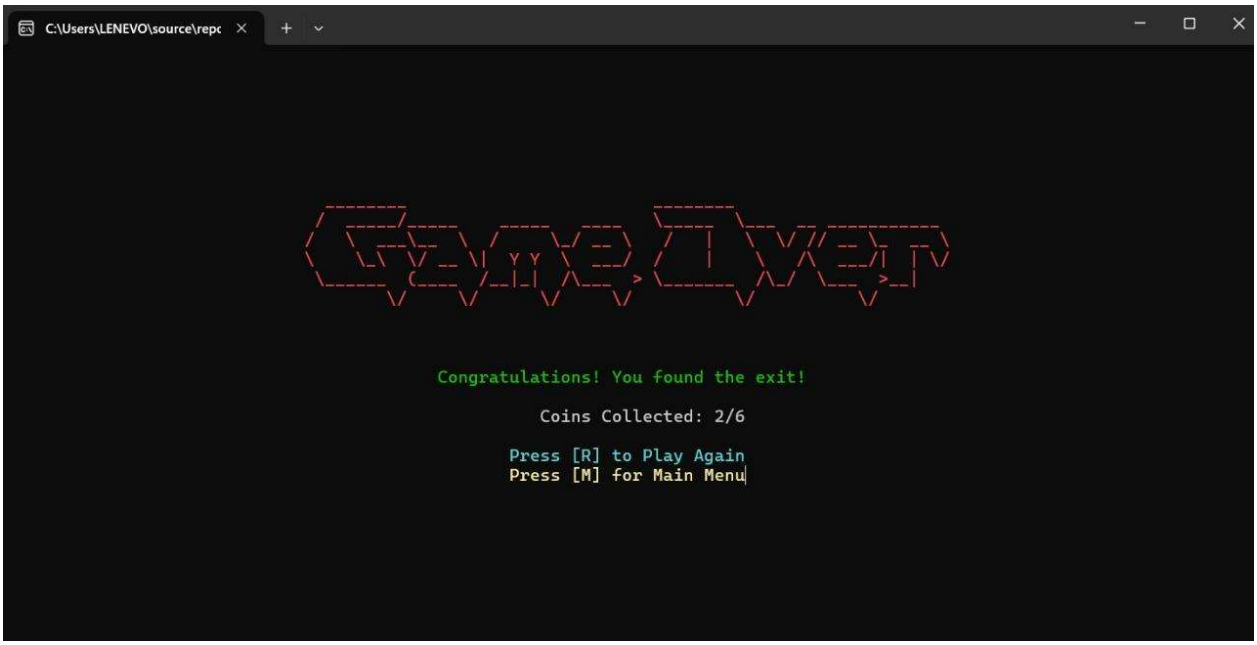
INVALID INPUT:

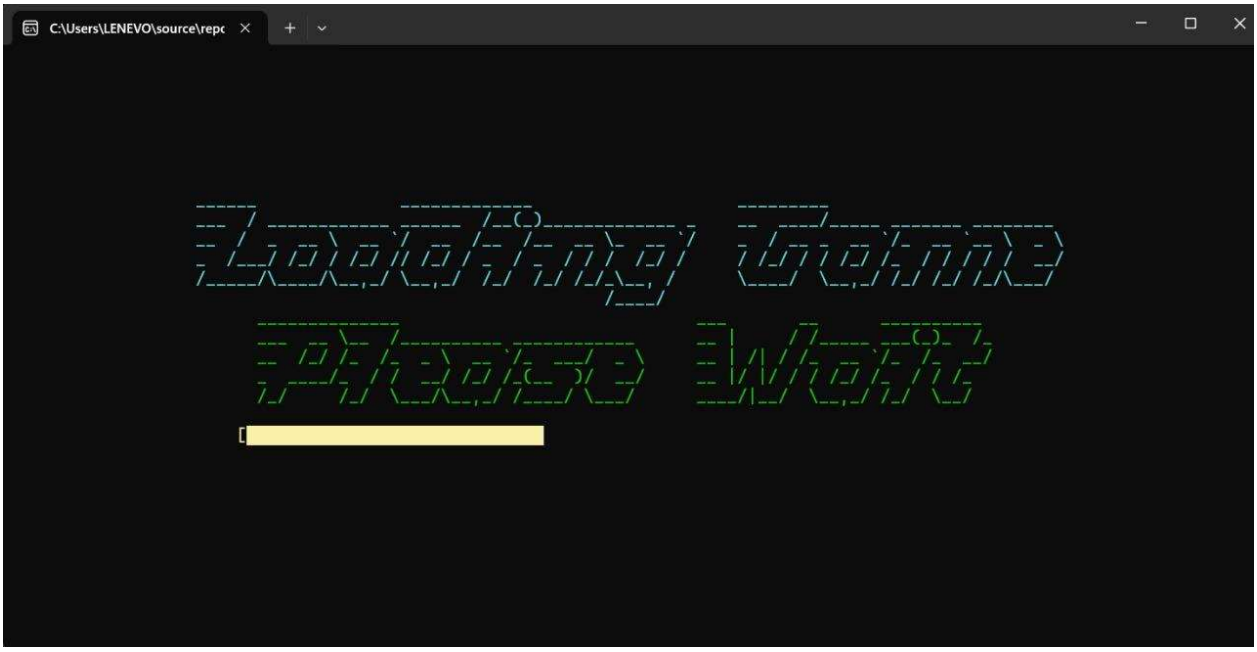

C:\Users\LENEVO\source\repc
GAME MENU
EXECUTE: MAZE_PROTOCOL.EXE
INITIATE: FLIGHT_SIMULATION.EXE
DECRYPT: WORD_MATRIX.EXE
TERMINATE: SESSION.EXE
ENTER COMMAND SEQUENCE (NUMBER): INVALID


C:\Users\LENEVO\source\repc
GAME MENU
EXECUTE: MAZE_PROTOCOL.EXE
INITIATE: FLIGHT_SIMULATION.EXE
DECRYPT: WORD_MATRIX.EXE
TERMINATE: SESSION.EXE
ENTER COMMAND SEQUENCE (NUMBER): 1


C:\Users\LENEVO\source\repc
WELCOME TO MAZE TREASURE HUNT!
Press R To Return To The Game Menu
or
Press P To Proceed To The Maze Game


C:\Users\LENEVO\source\repc
~MINI TREASURE MAZE HUNT~
Find the exit and collect coins (*)!
Coins: 2/6
Exit
Enter ->
Use ARROW KEYS to move, Q to quit to main menu


C:\Users\LENEVO\source\repc
Congratulations! You found the exit!
Coins Collected: 2/6
Press [R] to Play Again
Press [M] for Main Menu


C:\Users\LENEVO\source\repc


C:\Users\LENEVO\source\repc
Score: 1
--FLAPPY BIRD--
(>
)
GAME PAUSED! Press 'R' to resume or 'BACKSPACE' to quit


C:\Users\LENEVO\source\repc
Score: 1
--FLAPPY BIRD--
(>
)
GAME OVER!
Score: 1
Press R to replay or BACKSPACE to quit

C:\Users\LENEVO\source\repc

```
 _  _   __   __ _  ____  _  _  __  __ _
/ )( \ / _\ (  ( \ / ___)( \/ ) / _\ (  ( \
) __ (/    \/    / ( (_ \ \/ \/    \/    /
\_)(_/\_/\_/\_)__) \___/\_/\_/\_/\_/\_)__)
```

WELCOME, BRAVE GUESSER!

Ready to save a virtual life?

Guess the word before the stick figure meets its doom!

[1] - PLAY GAME

[2] - INSTRUCTIONS

[3] - RETURN TO MAIN MENU

C:\Users\LENEVO\source\repc

=== HOW TO PLAY ===

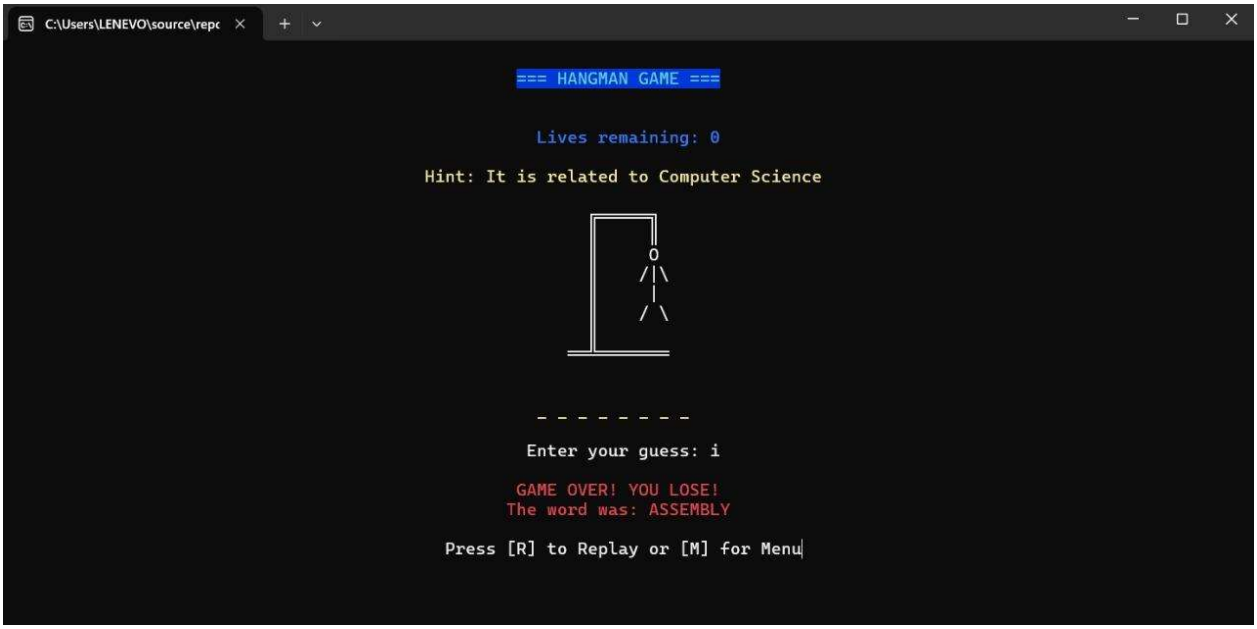
1. Guess letters one at a time

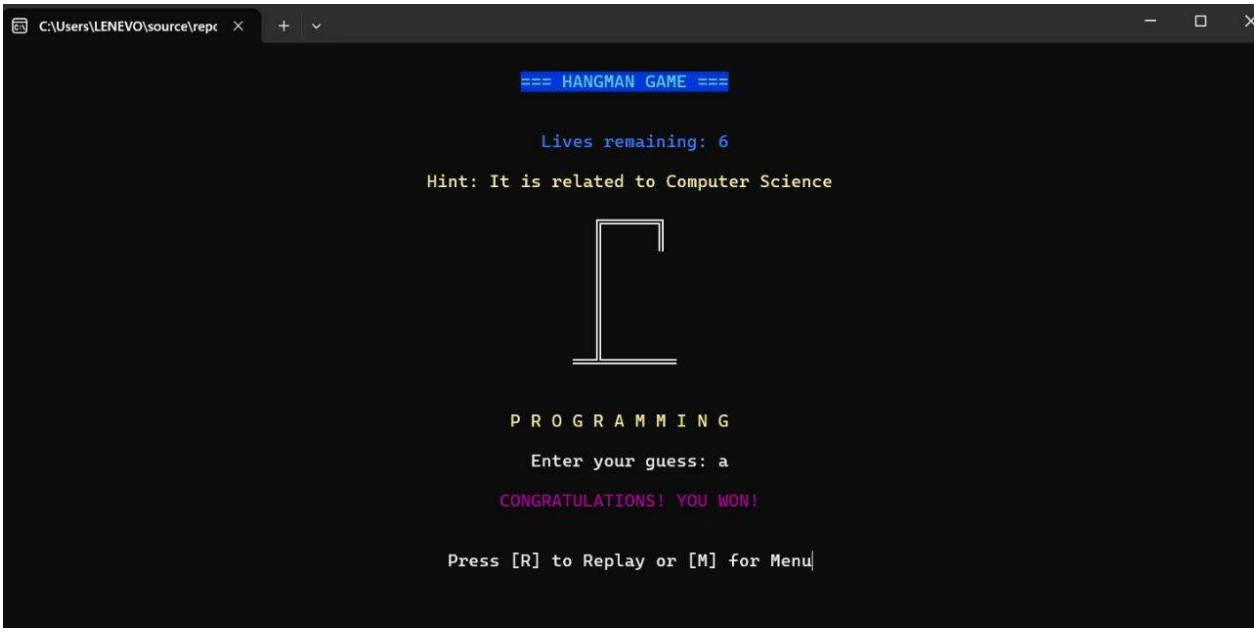
2. Each wrong guess adds a body part to the hangman

3. 6 wrong guesses and... GAME OVER!

4. Guess all letters correctly to WIN!

Press BACKSPACE to go back





Testing and Validation:

- Each game and feature is tested by each member of the group to ensure error free codes
- Hangman tested with different inputs repeatedly for smooth functioning
- Flappy bird tested with different movements with different intervals and timings
- Maze tested with valid and invalid moves for testing the exit logic
- Main menu tested for correct game selection and transitions into different games

7. Conclusion

Summary findings:

- All project objectives were met for all four games
- Each game illustrates fundamental COAL concepts: registers, loops, conditions, memory handling, and user input.
- The final versions of Hangman, Flappy Bird, Maze, and Main Menu worked reliably.

This project demonstrated how low-level programming affects both flow of control and performance.

Implementation reinforced understanding of how Assembly operations relate to the real behavior of a program.

Final remarks:

Implementing OOP concepts using assembly language highlighted the need for careful and precise coding

Assembly debugging strengthened the dry runs and understanding of memory usage

This project improved problem-solving skills and implementing graphics using assembly