National University of Computer & Emerging Sciences

Karachi Campus

HANGMAN GAME

Project Proposal

COAL LAB

Section: B

Group Members:

23k-0067 (Tanisha)

23i-2560 (emaan Arshad)

Project Proposal

- **Introduction**

The goal of this project is to develop a **Hangman Game** using Assembly language with the Irvine32 library. Hangman is a word-guessing game where players must guess letters of a hidden word. The objective is to implement this game in low-level assembly language, emphasizing the use of interrupts, memory management, and efficient input/output handling. This project will demonstrate core assembly concepts while delivering an engaging game experience.

- **Existing System**

While Hangman is a well-known game with many implementations in high-level languages such as Python, C, or Java, there are very few implementations in assembly language. Most existing systems leverage modern programming languages for ease of development, focusing on more sophisticated user interfaces and graphics. Existing versions of the game focus heavily on user-friendliness and advanced features like multiplayer and online play, which are beyond the scope of assembly programming.

- **Problem Statement**

Most implementations of the Hangman game are developed using high-level languages due to their simplicity in handling string manipulation and input/output operations. However, there is a lack of understanding of how such games can be developed at a lower level using assembly language. This project intends to bridge that gap by demonstrating how to implement core gameplay mechanics, user input, and game logic in Assembly with Irvine32.

Additionally, implementing this game in assembly introduces challenges such as limited string handling, complex memory management, and manual control over I/O, which high-level languages handle more easily.

- **Proposed Solution**

  This project will implement the Hangman game using Assembly language and the Irvine32 library. The game will display a series of underscores representing the hidden word and allow the player to guess letters. For each incorrect guess, a part of the hangman figure will be drawn. If the player guesses all the letters correctly before the hangman is fully drawn, they win; otherwise, they lose. Key features like tracking missed guesses, checking for already-guessed letters, and providing game-ending conditions will be included. By focusing on the core logic and user interactions, this project will demonstrate efficient programming techniques in Assembly language.

- **Salient Features**

  a) The player can input guesses one letter at a time.
  b) The player has up to 6 incorrect guesses before losing the game
  c) The game displays underscores (_) for each letter in the word and reveals correctly guessed letters in their respective positions.
  d) A simple hangman figure is drawn based on the number of incorrect guesses.
  e) The game will provide feedback when the word is fully guessed or when the 6 incorrect attempts have been exhausted.
  f) The player wins if they guess the word correctly and loses if the hangman is fully drawn

- **Tools & Technologies**

  a) **Programming Language:** Assembly Language (MASM - Microsoft Macro Assembler)
  b) **Library:** Irvine32 library (for basic input/output, random number generation, and other utilities)
  c) **Development Environment:** Microsoft Visual Studio 2019 with MASM setup
  d) **Operating System:** Windows (required for the Irvine32 library)