***Artificial Intelligent***



**Supervisor:**

Mr. Rebaz Najeeb

**Submission Date:**

August 3, 2024

Aryan Bestun

Hera Sabir

**Objective of the Project:**

The primary objective of this project was to implement a console-based Hangman game in Python. Hangman is a classic word-guessing game where players attempt to guess a hidden word by suggesting letters within a limited number of attempts. The aim was to create an interactive and user-friendly game that includes visual representation of the hangman stages, tracks user guesses, and provides feedback based on the correctness of the guesses.

**Methods:**

Word Selection: A list of possible words is defined, and a word is randomly selected from this list for the player to guess.

Game Loop: The game uses a while loop to manage the flow, allowing the player to continue guessing until they either guess the word correctly or run out of attempts.

Input Validation: Player inputs are validated to ensure they are single alphabetic characters.

Tracking Guesses: Correct and incorrect guesses are tracked separately. Correct guesses update the display of the word, while incorrect guesses reduce the number of remaining attempts.

Visual Representation: Different stages of the hangman are displayed based on the number of incorrect attempts, providing a visual cue to the player.

**Tools:**

Python: The primary programming language used for the implementation.

Random Module: Used to randomly select a word from the predefined list.

Challenges Faced and Overcoming Them

Input Validation: Ensuring that the player's input is a single alphabetic character was crucial to avoid errors. This was handled by checking the length of the input and whether it is alphabetic.

Solution: Added a condition to validate the input and prompt the user again if the input was invalid.

Tracking and Displaying Guesses: Keeping track of the guessed letters and updating the displayed word correctly posed a challenge.

Solution: Used a list to store the guessed word with underscores and updated this list as correct guesses were made. A set was used to track all guessed letters to avoid duplicate guesses.

Visual Representation: Creating and updating the visual stages of the hangman based on incorrect guesses required careful design.

Solution: Defined a list of strings representing each stage of the hangman. The appropriate stage was displayed based on the number of remaining attempts.

**Summary of Results**

The project successfully resulted in a fully functional Hangman game that can be played in the console. Key features include:

Random Word Selection: Each game starts with a randomly selected word from a predefined list.

Interactive Gameplay: Players can guess letters, and the game provides immediate feedback on whether the guess is correct or incorrect.

Visual Feedback: The hangman stages are displayed graphically, changing with each incorrect guess.

Input Validation: Ensures that only valid inputs are accepted, enhancing the user experience.

Winning and Losing Conditions: The game ends with a win if the player guesses the word correctly within the allowed attempts or a loss if they run out of attempts.