Hangman Game

Group-3

Faieq Daiyan Islam   
Department of Electrical & Computer Engineering   
North South UniversityDhaka, Bangladesh  
faieq.islam.241@northsouth.edu

Taijul Islam Siam  
Department of Electrical & Computer Engineering   
North South UniversityDhaka, Bangladesh  
taijul.siam.241@northsouth.eduSopnil Ghosh  
Department of Electrical & Computer Engineering  *)*  
North South UniversityDhaka, Bangladesh  
sopnil.ghosh.241@northsouth.edu

*Abstract*—This report presents the design and implementation of a console-based Hangman game written in the C programming language. The game involves players attempting to guess a randomly selected word by suggesting individual letters. The system provides feedback on correct or incorrect guesses while displaying the current state of the word and the hangman figure. The primary goal is to successfully guess the word before exhausting a limited number of attempts. The game is designed with user input validation, randomized word selection, and a visually appealing hangman display.

Keywords—Hangman game, C programming, user interaction, game mechanics.

# INTRODUCTION

The Hangman game is a classic word-guessing game that has entertained players for decades. This implementation demonstrates the effective use of control structures, randomization, and user input handling in C. The purpose of this project is to reinforce fundamental programming concepts such as arrays, string manipulation, conditional logic, and loops while providing an engaging interactive experience.

# SYSTEM DESIGN

## Random Word Selection

The game uses an array of predefined words (words[]) and corresponding hints (hints[]).A random word is selected using the rand() function seeded with the current time. The selected word and its corresponding hint are copied into buffers using strcpy().

## Hangman Display

The drawhangman() function dynamically updates the hangman directly based on the attempts remaining. It utilizes switch case statements to print different stickman representations of the hangman figure, progressively completing it.

## User Input Handling

The player input is collected using scanf(). Input validation ensures that only single alphabetic characters are accepted, enhancing the user experience. Non-alphabetic characters are handled appropriately using the isalpha() function. Multiple character inputs are handled using a certain while loop - while ((c = getchar()) != '\n' && c != EOF); inside an if statement. The semicolon at the end is used on purpose. It indicates that the loop has an empty body. This means that the loop will repeatedly execute the condition without performing any additional operations inside the loop itself.

# GAME LOGIC

## Word Guessing Mechanism

The game represents the unguessed letters using an array called guessedword[] initialized with underscores (‘\_’). Whenever a correct letter is guessed, the corresponding underscore is replaced with the correct letter.

## Win/Lose Logic

The player wins when all the correct letters are guessed (fullyguessed==1) and loses when all the attempts are used up (attemptsleft=0). The system provides feedback for both outcomes, displaying a congratulatory message for a successful attempt and a game over screen, correct word and the diagram of a fully drawn hangman for the failed attempt.

# RESULTS AND DISCUSSION

## Evaluation of Program Functionality

The Hangman game successfully meets its design objectives, providing a fully functional word-guessing experience. The stickman representation is clear and enhances the visual appeal of the game. Input validation ensures a smooth and error-free user experience, and the random word selection guarantees replayability.

## Challenges and Solutions

To ensure randomness srand(time(NULL)) was used. Input validation and player feedback was implemented to prevent invalid inputs. Hints and dynamic visuals were added for progress tracking.

## Future Enhancements

Adding tolower() will allow guesses to match regardless of case. Difficulty modes can be added by adding three different word and hint arrays for three sets of difficulties. Adding a restart feature would enhance user engagement.

# CONCLUSION

This report has detailed the design and implementation of a console-based Hangman game in C. The project demonstrates fundamental programming skills such as loops, conditionals, string manipulation, and user input validation. The game is both educational and entertaining, serving as a practical example of how simple algorithms can be combined to create an interactive application.

##### Acknowledgment

The authors would like to thank the guidance of their peers and various programming communities and resources that contributed to the successful completion of this project.