Hanging Man Game

GROUP 19

Johnest Wheyne D. Ogania

Christian Clark Gatil

Course/Year: BSCS - 1st Year Section C104

Submission Date: 1st Week of December

Program Description:

The hanging Man is a simple game, mainly functioned by C language codes. Where in you are able to input the words with clues that will be stored within the game. The game ends if you are unable to guess the words letter within 6 times and the stick figure man hanging will show up.

Program Requirements:

Your program must include:

- At least one function
- At least one array
- At least one iteration/loop statement
- Proper input and output
- At least one conditional statement

Code Implementation:

The code for the Hanging Man Game is shown below. Ensure the code is properly formatted and commented for clarity.

```
/*****************************************************
 * Project Title: Hanging Man
 * Author: Johnest Wheyne D. Ogania & Christian Clark Gatil
 * Date: 1st Week of December
 * Description: A simple program of Hanging man
*****************************************************/

#include <stdio.h>

#include <string.h>
```

```
#include <stdlib.h>

char toUpperCase(char letter);
int isLetterInWord(char letter, const char *word);
void displayHangman(int wrongGuesses);
int calculateScore(int wrongGuesses);
int loadWordsAndClues(char words[][50], char clues[][100], int *count);
void addWordAndClue();
void playGame();

char toUpperCase(char letter) {
    if (letter >= 'a' && letter <= 'z') {
        return letter - ('a' - 'A');
    }
    return letter;
}

int isLetterInWord(char letter, const char *word) {
    for (int i = 0; word[i] != '\0'; i++) {
        if (toUpperCase(word[i]) == toUpperCase(letter)) {
            return 1;
        }
    }
    return 0;
}
```

```
void displayHangman(int wrongGuesses) {
    const char *hangman[] = {
        "  _______\n"
        "  |     |\n"
        "  |\n"
        "  |\n"
        "  |\n"
        "  |\n"
        "_|_\n",

        "  _______\n"
        "  |     |\n"
        "  |     O\n"
        "  |\n"
        "  |\n"
        "  |\n"
        "_|_\n",

        "  _______\n"
        "  |     |\n"
        "  |     O\n"
        "  |     |\n"
        "  |\n"
        "  |\n"
        "_|_\n",
```

```
" _______\n"
" |     |\n"
" |     O\n"
" |    /|\n"
" |\n"
" |\n"
"_|_\n",

" _______\n"
" |     |\n"
" |     O\n"
" |    /|\\\n"
" |\n"
" |\n"
"_|_\n",

" _______\n"
" |     |\n"
" |     O\n"
" |    /|\\\n"
" |    /\n"
" |\n"
"_|_\n",

" _______\n"
" |     |\n"
```

```
        "  |     O\n"
        "  |    /|\\\n"
        "  |    / \\\n"
        "  |\n"
        "_|_\n"
    };

    printf("%s\n", hangman[wrongGuesses]);
}

int calculateScore(int wrongGuesses) {
    if (wrongGuesses == 0) {
        return 100;
    } else if (wrongGuesses <= 2) {
        return 90;
    } else if (wrongGuesses <= 4) {
        return 70;
    } else {
        return 50;
    }
}

int loadWordsAndClues(char words[][50], char clues[][100], int *count) {
    const char *defaultWords[] = {
        "WHALE", "AURORA", "WOODY", "PHILIPPINES", "CHEETAH",
        "SATURN", "EMERALD", "SUBWAY", "PIANO", "PENGUIN"
```

```
};
const char *defaultClues[] = {
    "A mammal that lives in water",
    "A Disney princess",
    "A character from the movie Toy Story",
    "A country in Asia",
    "A fast animal",
    "A planet in the Solar System",
    "A type of gemstone",
    "A transportation without being in the streets",
    "A musical instrument",
    "A flightless bird"
};

*count = 10;
for (int i = 0; i < 10; i++) {
    strcpy(words[i], defaultWords[i]);
    strcpy(clues[i], defaultClues[i]);
}

FILE *file = fopen("words.txt", "r");
if (!file) {
    return 1;
}

while (fscanf(file, "%49[^|]|%99[^\n]%*c", words[*count], clues[*count]) != EOF) {
```

```
        (*count)++;
    }
    fclose(file);
    return 1;
}

void addWordAndClue() {
    char word[50], clue[100];
    printf("Enter the new word: ");
    scanf("%s", word);
    getchar();
    printf("Enter the clue for the word: ");
    fgets(clue, 100, stdin);
    clue[strcspn(clue, "\n")] = 0;

    FILE *file = fopen("words.txt", "a");
    if (!file) {
        printf("Error: Could not open file to save word.\n");
        return;
    }
    fprintf(file, "%s|%s\n", word, clue);
    fclose(file);

    printf("Word and clue added successfully!\n");
}
```

```
void playGame() {
    char words[100][50], clues[100][100];
    int wordCount;

    if (!loadWordsAndClues(words, clues, &wordCount)) {
        printf("Error: No words and clues available. Add some first.\n");
        return;
    }

    int wordIndex = rand() % wordCount;
    const char *word = words[wordIndex];
    const char *clue = clues[wordIndex];

    int wordLength = strlen(word);
    char guessed[wordLength + 1];
    char guessedLetters[26] = {0};
    int guessedLetterCount = 0;

    for (int i = 0; i < wordLength; i++) {
        guessed[i] = '_';
    }
    guessed[wordLength] = '\0';

    int maxGuesses = 6;
    int wrongGuesses = 0;
```

```
printf("\nClue: %s\n", clue);
printf("The word has %d letters.\n", wordLength);

while (wrongGuesses < maxGuesses) {
    printf("\nCurrent word: ");
    for (int i = 0; i < wordLength; i++) {
        printf("%c ", guessed[i]);
    }
    printf("\nRemaining guesses: %d\n", maxGuesses - wrongGuesses);
    printf("Guessed letters: ");
    for (int i = 0; i < guessedLetterCount; i++) {
        printf("%c ", guessedLetters[i]);
    }
    printf("\n");

    displayHangman(wrongGuesses);

    printf("Enter a letter: ");
    char guess;
    scanf(" %c", &guess);
    guess = toUpperCase(guess);

    int alreadyGuessed = 0;
    for (int i = 0; i < guessedLetterCount; i++) {
        if (guessedLetters[i] == guess) {
            alreadyGuessed = 1;
```

```
            break;
        }
    }

    if (alreadyGuessed) {
        printf("You already guessed the letter %c.\n", guess);
        continue;
    }

    guessedLetters[guessedLetterCount++] = guess;

    if (isLetterInWord(guess, word)) {
        printf("Good guess!\n");

        for (int i = 0; i < wordLength; i++) {
            if (toUpperCase(word[i]) == guess) {
                guessed[i] = word[i];
            }
        }

        if (strcmp(guessed, word) == 0) {
            printf("\nCongratulations! You've guessed the word: %s\n", word);
            printf("Your final score: %d\n", calculateScore(wrongGuesses));
            return;
        }
    } else {
```

```
            printf("Incorrect guess.\n");
            wrongGuesses++;
        }
    }

    displayHangman(wrongGuesses);
    printf("\nGame over! The word was: %s\n", word);
    printf("Your final score: 0\n");
}
int main() {
    char choice;

    do {
        printf("\n**********************************\n");
        printf("*       Welcome to Hangman!      *\n");
        printf("* 1. Start Game                   *\n");
        printf("* 2. Add Word and Clue            *\n");
        printf("* 3. Exit                         *\n");
        printf("**********************************\n");

        printf("Enter your choice: ");
        scanf(" %c", &choice);

        switch (choice) {
            case '1':
                playGame();
```

```
            break;
        case '2':
            addWordAndClue();
            break;
        case '3':
            printf("Goodbye! Thanks for playing!\n");
            break;
        default:
            printf("Invalid choice. Try again.\n");
    }
  } while (choice != '3');

  return 0;
}
```

Explanation of Components:

**Function: ** Displays a simple gaming code in where you are able to input a word and a clue of your own

**Array: ** Stores the words and clues that user have inputted

**Iteration/Loop Statements: **

**Input/Output: ** Guess the letters of the word and when the word was guessed wrong the hanging man will show shortly

**Conditional Statements: ** Used for educational purposes to highlight the critical thinking of an individual

Sample Output:

When the program is executed, the following operations can be demonstrated:

- Displaying the font of the game which has the Start Game, add word and clue, and exit.

- New words will be stocked within the game if there was any inputted

- The hanging man will show up bit by bit if the letter was guessed wrong

Reflection:

While coding the game I found it challenging to completely change an code once you have run it and it has completely functioned well. It is hard to revise the codes and find the error completely. However, these challenges has made me grow as an individual,nd I have learned to use arrays and functions to arrange the code correctly.