**Project 1: Hangman Game**

**Brittany Chan**

**CIS-5 47106**

**May 7, 2023**

Table of Contents

[Background of the Game 3](#_Toc136870947)

[How the Game Works 3](#_Toc136870948)

[Purpose of Every Version Made 3](#_Toc136870949)

[Gameplay 5](#_Toc136870950)

[Pseudo Code 7](#_Toc136870951)

[Flowchart 10](#_Toc136870952)

Introduction

# **Background of the Game**

This is a hangman game that allows the user to play solo or multiplayer. The user will be prompted to select whether they want to play solo or multiplayer; picking the former will cause the program to refer to an already made file with a list of words and will pick a random word from the list. The latter will cause the program to generate a random number that will be the length of the word that the user needs to input. For example, the program will state “Enter a word that is 6 letters long.” The aim of the game is for the user to have fun guessing words and learn new words through the random generated word file.

# **How the Game Works**

**Objective of the Game:** Guess the word before completing the image of the hangman.

**Rules of the Game:**

1. Choose the mode of player
2. Enter a letter to guess
3. Avoid getting unnecessary deductions by inputting anything other than a letter
4. Win the game by guessing the word by entering the right letters and before you reach 6 strikes

Versions

Version 1 tests a lot of things like how the hangman should be displayed and focuses more on formatting. Version 2 focuses on whether the loops are working correctly and if the conditional statements are printing out when they are expected to. Version 3 attempts to use bools and switch statements to control the outputs. There are also other files in the Project 1 folder that serve as my thought process as well as tests to see whether individual parts of my code work.

# **Purpose of Every Version Made**

1. Project\_1\_v1
   1. The very first implementation of the game and only has primitive code such as initializing the game and manually assigning a char variable to each letter in the word due to not being able to add advanced concepts such as arrays yet.
   2. This version also only accepts 1 player.
2. Project\_1\_v2
   1. Added a choice to play with 2 players.
   2. Implemented for loops, do-while loops for the functioning of the game.
   3. Implemented a primitive way to check whether the guess is correct by using a for loop to find if the guessed letter appears in the word.
   4. Displays the picture of the hangman depending on the number of strikes made.
3. Project\_1\_v3\_SwitchStatements
   1. A switch statement depending on the status.
   2. A status of true means the guess was wrong, and false if the answer is correct
   3. After checking the status, there is an if statement to check how many strikes have been made and will display the hangman based on that number.
4. Project\_2\_Functions
   1. Starting to create functions in place of code in the main function.
   2. This version only tried to create functions for initializing the game such as entering the number of players and the word.
5. Project\_1\_MoreFunctions
   1. This version included most of the functions needed for the game such a function to verify the answer, modifying the scores, outputting the hangman, outputting the wrong guesses.
   2. Fixed opening and reading of files and made sure that the word generated works.
6. Project\_Revisions
   1. Needed to make sure that adding to the vector worked correctly.
7. Project\_Revisions2
   1. Modifying the blanks when the inputted guess is correct.
8. Project\_Revisions\_Final
   1. Implemented bubble and selection sort in sorting all the inputted guesses alphabetically.
   2. Outputting all the guesses for the player’s reference
   3. Creating an answer key to show the user all the guesses they made and marking them with ‘C’ for correct and ‘W’ for incorrect.
   4. Allows player/s to play again

The Game

# **Gameplay**

For 1 player

A close-up of black text

Description automatically generated with low confidence

For 2 players

A picture containing text, font, white, receipt

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

A screenshot of a phone

Description automatically generated with medium confidenceA screenshot of a phone

Description automatically generated with low confidence

Lost Won

Logic of the Game

# Pseudo Code

*Initialize*

*Score = 6*

*Strike = 0*

*Correct = 0*

*Key[rowMax][colMax]*

*Init()*

*Greet players*

*Prompts user if they want to play with 1 player or 2 players*

*getChc()*

*Accept user input*

*setPlyr()*

*If 1 player,*

*Choose random word from word list*

*Else*

*Require 2nd player to input a word of the required length*

*Return word*

*lnthChk()*

*while (word.length() != required length)*

*Prompt user to enter a word with appropriate length*

*Ask user if ready to start*

*Return choice*

*gmDsply()*

*If choice is yes,*

*Display hangman and blanks*

*Else*

*Exit program*

*While strike != 6 && correct != word.length()*

*guess()*

*Prompt user to take a guess*

*Input answer*

*chckAns()*

*If answer is within range of capital and lowercase letters*

*For letter in word*

*If answer == letter in word*

*Status = false*

*Increment correct*

*Else*

*Prompt user to enter a valid answer*

*If status == false*

*Score += 5*

*Prompt “That was a nice guess”*

*Else*

*Score = sqrt(score)*

*Strike++*

*Return status*

*If status == true*

*Push back answer to Wrong Answers vector*

*gmStat()*

*Output Score*

*Output Strike*

*Output Wrong Answers*

*If status == true*

*Bubble sort Wrong Answers*

*Else*

*Selection sort Wrong Answers*

*Display Hangman based on number of Strikes*

*Output all 2D array Answer key*

*If letter in Answer key is in word*

*Output ‘C’ for correct*

*Else*

*Output ‘W’ for wrong*

*If correct == word.length()*

*Output ‘Congratulation’s and Score*

*If strike == 6*

*Output ‘You died’ and Score*

*Output ‘The word was’ word*

*Prompt ‘Would you like to play again?’*

*If choice == yes*

*Restart program*

*Else*

*‘Thank you for playing’*

*Return 0*

# Flowchart























