**Assignment: Guessing Game in Bash**

**Introduction**

This assignment is designed to test your skills in using Git and GitHub, creating makefiles, and writing Bash scripts. The objective is to create a simple guessing game that prompts the user to guess the number of files in the current directory.

**Program: guessinggame.sh**

**Purpose**

The guessinggame.sh script will continuously prompt the user to guess the number of files in the current directory. It will provide feedback on whether the guess is too high or too low, and congratulate the user upon a correct guess.

**Requirements**

* The script should count the number of files in the current directory.
* It should prompt the user for a guess.
* Provide feedback on the guess (too high, too low).
* Congratulate the user when they guess correctly.

**Implementation**

Here is a sample implementation of the guessinggame.sh script:

#!/bin/bash

# Count the number of files in the current directory

file\_count=$(ls -1 | wc -l)

# Initialize the user's guess

guess=-1

# Loop until the user guesses correctly

while [[ $guess -ne $file\_count ]]; do

# Prompt the user for a guess

read -p "Guess the number of files in the current directory: " guess

# Check if the guess is a number

if ! [[ "$guess" =~ ^[0-9]+$ ]]; then

echo "Please enter a valid number."

continue

fi

# Provide feedback on the guess

if [[ $guess -lt $file\_count ]]; then

echo "Too low! Try again."

elif [[ $guess -gt $file\_count ]]; then

echo "Too high! Try again."

fi

done

# Congratulate the user

echo "Congratulations! You guessed the correct number of files: $file\_count."

**Explanation of the Code**

- The script starts by counting the number of files in the current directory using ls -1 | wc -l.

- It initializes a variable guess to -1 to ensure the loop starts.

- A while loop continues until the user guesses the correct number of files.

- The read command captures user input.

- The script checks if the input is a valid number and provides appropriate feedback.

- Once the correct guess is made, the user is congratulated.

**Conclusion**

This assignment demonstrates the ability to create a simple interactive Bash script that engages the user in a guessing game. It reinforces concepts such as loops, conditionals, and user input handling in Bash scripting.