#!/bin/bash

# This is an age guessing game

# Author: Xuyu Zhang, Student Number: 10493476

echo "######################"

echo "# #"

echo "# Guess Age Game #"

echo "# #"

echo "######################"

# Using the while loop, if someone finds that the program entered is wrong, press 0 to immediately exit.

# Also, the player can press 1 to continue playing the game.

while true; do

echo "1: Start"

echo "2: Exit"

#Tell the player to type 0 or 1.

read -p "Please enter 0 or 1: " num

# When the player presses 1, the game will start.

if [[ $num -eq 1 ]]; then

# This step is order to create a number between 20 and 70.

age=$[RANDOM%50+20]

# Explain the rules of this game.

echo "There is a man between the ages of 20 and 70, try to guess his age!"

# Using the while loop, the player is allowed to rewrite the number after guessing the wrong age until the guess is correct.

# Also, this is allowed to fill in the wrong input.

while true; do

# Guide the player to enter numbers to play this game.

read -p "Please enter the number of age you guess: " guessage

# If the player enters a number between 20 and 70, the game continues.

if [[ $guessage -ge 20 ]] && [[ $guessage -le 70 ]]; then

# If the guess is less than the correct number, the prompt will appear and the player will be asked to enter the number again.

if [[ $guessage -lt $age ]];then

echo "Unfortunately, your guess is younger than that. Please try again."

# If the guess is greater than the correct number, the prompt will appear and the player will be asked to enter the number again.

elif [[ $guessage -gt $age ]];then

echo "Unfortunately, your guess is older than that. Please try again."

# If the guess is equal to the correct number, the prompt will tell the player that he has guessed correctly and then exit the game.

else

# Congratulate the player when he guesses the correct age and exit the game.

echo "Congratulations! He is $age years old!"

exit

fi

# If the player enters another number or text, an error is reported and the player is asked to fill it out again.

else

echo "Sorry, invalid number or word entered. Please try again."

fi

done

# When the player presses 0, the game will be over.

elif [[ $num -eq 0 ]]; then

exit

# If the player enters another number or text, an error is reported and the player is asked to fill it out again.

else

echo "Sorry, invalid number or word entered. Please try again."

fi

done

Output：