## Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".

#!/bin/bash

if [ -e "myfile.txt" ]; then

echo "File exists"

else

echo "File not found"

fi

# Assignment 2: Write a script that reads numbers from the user until they enter '0'. The script should also print whether each number is odd or even.

#!/bin/bash

while true; do

# Read input from the user

echo -n "Enter a number (0 to quit): "

read number

if [ "$number" -eq 0 ]; then

echo "Exiting..."

break

fi

# Check if the number is even or odd

if [ "$((number % 2))" -eq 0 ]; then

echo "$number is even"

else

echo "$number is odd"

fi

done

# Assignment 3: Create a function that takes a filename as an argument and prints the number of lines in the file. Call this function from your script with different filenames

#!/bin/bash

count() {

filename="$1"

if [ -f "$filename" ]; then

num=$(wc -l < "$filename")

echo "The number of lines in $filename is: $num"

else

echo "File '$filename' not found."

fi

}

count "file1.txt"

count"file2.txt"

# Assignment 4: Write a script that creates a directory named TestDir and inside it, creates ten files named File1.txt, File2.txt, ... File10.txt. Each file should contain its filename as its content (e.g., File1.txt contains "File1.txt").

#!/bin/bash

mkdir -p TestDir

# Navigate to the TestDir directory

cd TestDir || exit

# Create ten files named File1.txt, File2.txt, ... File10.txt

for i in {1..10}; do

filename="File${i}.txt"

echo "$filename" > "$filename"

done

echo "Files created successfully."

# Assignment 5: Modify the script to handle errors, such as the directory already existing or lacking permissions to create files.

# Add a debugging mode that prints additional information when enabled.

#!/bin/bash

DB=false

db() {

if [ "$DB" = true ]; then

echo "[DB] $1"

fi

}

create\_files() {

db "Creating files..."

for ((i=1; i<=10; i++)); do

filename="File${i}.txt"

db "Creating file: $filename"

echo "$filename" > "$filename"

if [ $? -ne 0 ]; then

echo "Error: Failed to create file $filename"

exit 1

fi

done

db "Files created successfully."

}

# Check if TestDir directory already exists

if [ -d "TestDir" ]; then

echo "Error: Directory 'TestDir' already exists."

exit 1

fi

# Create a directory named TestDir

mkdir -p TestDir

if [ $? -ne 0 ]; then

echo "Error: Failed to create directory 'TestDir'. Check permissions."

exit 1

fi

# Navigate to the TestDir directory

cd TestDir || exit

create\_files

echo "Task completed successfully."

# Assignment 6: Given a sample log file, write a script using grep to extract all lines containing "ERROR". Use awk to print the date, time, and error message of each extracted line.Data Processing with sed

#!/bin/bash

# Define the log file

log\_file="sample.log"

# Use grep to extract lines containing "ERROR" from the log file

error\_lines=$(grep "ERROR" "$log\_file")

# Use awk to print the date, time, and error message of each extracted line

echo "$error\_lines" | awk '{print $1, $2, $3, $0}' | sed -E 's/^[^ ]\* [^ ]\* [^ ]\* (.\*)/\1/'

# Assignment 7: Create a script that takes a text file and replaces all occurrences of "old\_text" with "new\_text". Use sed to perform this operation and output the result to a new file.

#!/bin/bash

# Check if correct number of arguments are provided

if [ "$#" -ne 2 ]; then

echo "Usage: $0 input\_file new\_text"

exit 1

fi

input\_file="$1"

new\_text="$2"

# Check if input file exists

if [ ! -f "$input\_file" ]; then

echo "Error: Input file '$input\_file' not found."

exit 1

fi

# Perform the replacement using sed and modify the input file directly

sed -i "s/old\_text/$new\_text/g" "$input\_file"

echo "Replacement completed in $input\_file."