**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".**

**Code:-** #!/bin/bash

# Specify the filename to check

filename="myfile.txt"

# Check if the file exists using test command

if test -f "$filename"; 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.**

**Code:-** #!/bin/bash

while true; do

# Get user input

read -p "Enter a number (0 to quit): " number\_str

# Check if input is '0' to quit

if [[ "$number\_str" == "0" ]]; then

break

fi

# Check for valid integer (using regex)

if ! [[ "$number\_str" =~ ^[-+]?[0-9]+$ ]]; then

echo "Invalid input. Please enter a number."

continue

fi

# Convert string to integer

number=$((number\_str))

# Check if even or odd

if (( number % 2 == 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.**

**Code:-** #!/bin/bash

# Function to count lines in a file

count\_lines() {

# Get the filename as argument

local filename="$1"

# Check if file exists

if ! [[ -f "$filename" ]]; then

echo "File not found: $filename"

return 1 # Indicate error (non-zero exit code)

fi

# Use wc command to count lines

number\_of\_lines=$(wc -l < "$filename")

# Print the result

echo "$filename has $number\_of\_lines lines"

}

# Call the function with different filenames

filenames=("myfile.txt" "data.csv" "image.jpg")

for filename in "${filenames[@]}"; do

count\_lines "$filename"

done

**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").**

**Code:-** #!/bin/bash

# Directory name

dir\_name="TestDir"

# Number of files

num\_files=10

# Check if directory exists (optional)

if [[ -d "$dir\_name" ]]; then

echo "Directory '$dir\_name' already exists. Skipping directory creation."

else

# Create the directory

mkdir -p "$dir\_name"

if [[ $? -eq 0 ]]; then

echo "Directory '$dir\_name' created successfully."

else

echo "Error creating directory '$dir\_name'."

exit 1 # Exit with error code

fi

fi

# Create files with content

for i in $(seq 1 $num\_files); do

filename="File$i.txt"

filepath="$dir\_name/$filename"

echo "$filename" > "$filepath" # Redirect content to file

if [[ $? -eq 0 ]]; then

echo "Created file: $filepath"

else

echo "Error creating file: $filepath"

fi

done

echo "All files created successfully (if no errors reported above)."

**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.**

**Code:-** #!/bin/bash

# Log file path (replace with your actual log file path)

log\_file="sample.log"

# Extract lines with "ERROR" using grep and pipe to awk

grep "ERROR" "$log\_file" | awk '{print $1, $2, $3, $4, substr($0, index($0, $5))}'

# Explanation of awk command:

# - $1, $2, $3: Capture date, time, and first part of the message (fields may vary based on your log format)

# - index($0, $5): Find the starting position of the fifth word (error message)

# - substr($0, index($0, $5)): Extract the error message from the entire line starting at the fifth word