Assignment-6

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"".

Solution:-

```
#!/bin/bash

filename="myfile.txt"

if [ -e "$filename" ];

then

echo "file exists"

else

echo "filen 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.

Solution:-

```
#!/usr/bin/bash

while true; do

echo -n "Enter a number (0 to quit):" read number if [ "$number" -eq 0 ]; then echo "Exiting.."
break

fi then

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

else

echo "$number is even"

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.

Solution:-

```
Count_lines() {
Filename=”$1”
If [ -f “$filename” ]; then
Lines=$(wc -1 < $filename”)
Echo “Number of lines in $filename : $lines”
Else
Echo “$filename does exit”
Fi
}
Count_lines”myfile1.txt”
Count_lines”myfile2.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"").

Solution:-

```
#!/bin/bash

# Create directory
directory_name="TestDir"
mkdir -p $directory_name

# Create files with content
for i in {1..10}
do
  filename="File$i.txt"
  filepath="$directory_name/$filename"
  echo $filename > $filepath
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.

Solution:-

```
#!/bin/bash

# Enable debugging mode if the DEBUG variable is set to true
DEBUG=false

# Function to print debug messages
debug() {
  if [ "$DEBUG" = true ]; then
    echo "DEBUG: $1"
  fi
}

# Set the directory name
directory_name="TestDir"

# Check if the directory already exists
if [ -d "$directory_name" ]; then
  echo "Error: Directory '$directory_name' already exists."
  exit 1
else
  # Attempt to create the directory
  mkdir -p $directory_name
  if [ $? -ne 0 ]; then
    echo "Error: Failed to create directory '$directory_name'. Check your permissions."
    exit 1
  fi
  debug "Directory '$directory_name' created successfully."
fi

# Create files with content
for i in {1..10}; do
  filename="File$i.txt"
  filepath="$directory_name/$filename"
  echo $filename > $filepath
  if [ $? -ne 0 ]; then
    echo "Error: Failed to create file '$filepath'. Check your permissions."
    exit 1
  fi
  debug "File '$filepath' created with content '$filename'."
done

echo "Files created 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

Solution:-

```
#!/bin/bash

# Check if the log file is provided as an argument
if [ "$#" -ne 1 ]; then
  echo "Usage: $0 <logfile>"
  exit 1
fi

logfile=$1

# Check if the log file exists
if [ ! -f "$logfile" ]; then
  echo "Error: File '$logfile' not found."
  exit 1
fi

# Use grep to extract lines containing "ERROR" and use awk to process the lines
grep "ERROR" "$logfile" | awk '{ print $1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11 }' | sed
's/INFO/Information:/g'

# If further processing is needed, use sed here (example: formatting error messages)
# Example sed command to remove redundant spaces (this is just an example and can be modified
as needed)
# sed 's/  */ /g'

echo "Error extraction and processing completed."
```

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.

Solution:-

```
#!/bin/bash

# Check if the correct number of arguments is provided
if [ "$#" -ne 3 ]; then
  echo "Usage: $0 <input_file> <old_text> <new_text>"
  exit 1
fi

# Assign arguments to variables
input_file=$1
old_text=$2
new_text=$3
output_file="output_${input_file}"

# Check if the input file exists
if [ ! -f "$input_file" ]; then
  echo "Error: File '$input_file' not found."
  exit 1
fi

# Use sed to replace old_text with new_text and output to a new file
sed "s/${old_text}/${new_text}/g" "$input_file" > "$output_file"

echo "Text replacement completed. Output saved to '$output_file'."
```