Assignment-1

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

File3="myfile.txt"


if [ -f "$file3" ]; then

   echo "File exists"

else

   echo "File not found"

fi
```

Assignment-2

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:

```
#!/bin/bash

while true; do

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

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

                  echo "Exiting---"

                  break

         fi

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

          then
```

```
                echo "$number is even."
         else
               echo "$number is odd."
        fi
done
```

Assignment-3

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:

```
#!/bin/bash
count_lines() {
        local file3=$1
        local num_lines=$(wc -l  < "$file3")
        echo "Number of lines in $filename: $num_lines"
}
count_lines "file3.txt"
count_lines "file1.txt"
```

Assignment-4

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
```

```
mkdir TestDir

cd TestDir || exit


for ((i=1; i<=10; i++))
do
    filename="File${i}.txt"
    echo "$filename" > "$filename"
    echo "Created $filename with content \"$filename\""
 done

echo "Files created successfully in TestDir."
```

Assignment-5

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

Set  -x
Directory=”wipro”
If [  -d “$directory”  ]
then
      echo  “Directory exists.”
```

```
else
        mkdir –p  “$directory”
        echo “Directory created.”
fi
set +x
```

Assignment-6

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


#sample log file path
Logfile=”sample.log”


#use grep to extract lines containing “ERROR” and pass it to awk for processing


Grep “ERROR”  “$logfile” | \
Awk ‘{
         #Extract date and time
         Date_time = $1 “  “  $2


     #Remove date and time from the original line
       $1=$2=””
      #print date, time, and the rest of the line (error message)
```

```
    Print date_time, $0
}’
```

```
sample.log
-------------------
2024-05-16 08:30:15 INFO: Application started
2024-05-16 08:31:22 ERROR: Database connection failed
2024-05-16 08:32:45 WARNING : Disk space low
2024-05-16 08:33:12 ERROR: Invalid input received
2024-05-16 08:34:55 ERROR: Server crashed
```

Assignment-7

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 two arguments are provided
if [ $# -ne 2 ]; then
  echo "Usage: $0 <input_file> <output_file>"
  exit 1
fi
```

```
# Extract arguments
input_file="$1"
output_file="$2"

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

# Perform sed operation and redirect output to new file
sed 's/old_text/new_text/g' "$input_file" > "$output_file"

# Inform user
echo "Replaced 'old_text' with 'new_text' in '$input_file' and saved the result
to '$output_file'."
```