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".

```
name= "file.txt"
if [ -f "$name" ];
then
        echo "File found"
else
        echo "File not found"
fi
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment1.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cat assignment1.sh
name="file.txt"
if [ -f "$name" ];
then
        echo "File found"
else
        echo "File not found"
fi

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ls
assignment1.sh  file.txt

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment1.sh
File found
```

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.

```
echo "Exit code 0"
condition=true
while $condition
do
        echo "Enter number:"
        read number
if((number > 0))
then
        if((number%2==0))
        then
                echo "$number is Even"
        else
                echo "$number is odd"
        fi
else
        condition=false
fi
done
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment2.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment2.sh
Exit code 0
Enter number:
3
3 is odd
Enter number:
4
4 is Even
Enter number:
0
```

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.

```
lines=$( wc -l < file.txt )

echo "Number of lines present in file.txt is $lines"
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment3.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cat file.txt
Task:
Find a number of
lines present
in file

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment3.sh
Number of lines present in file.txt is 4
```

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").

```
mkdir TestDir

for i in $(seq 1 10);

do
        touch TestDir/File$i.txt

        echo "File$i.txt" > TestDir/File$i.txt

done
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment4.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment4.sh
mkdir: cannot create directory ‘TestDir’: File exists

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cd TestDir

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell/TestDir
$ ls
File1.txt   File2.txt  File4.txt  File6.txt  File8.txt
File10.txt  File3.txt  File5.txt  File7.txt  File9.txt

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell/TestDir
$ cat file1.txt
File1.txt

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell/TestDir
$ cat file10.txt
File10.txt
```

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.

```
cd(){
  local path=$file1
  local debug=$file2

  if [ "$debg" == "true" ];
then
        echo "creating directory: $path"
fi

  mkdir -p "$path" 2>/dev/null
  local status=$?

  if [ $status -eq 0 ];
then
    if [ "$debug" == "true" ];
     then
                echo "succussfully crested directory: $path"
        fi

    elif [ -d "$path" ];
    then
        if [ "$debug" == "true" ]
        then
        echo "Directory alredy epresent: $path"
        fi
        echo "Error: directory alredy present: $path"
    else
        if [ "$debug" == "true" ];
        then
                echo "error occured"
        fi
        echo "Error unable to create directory"
    fi
}
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment5.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment5.sh
Error unable to create directory
```

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

```
FILE="sample.log"

grep "ERROR" "$LOG_FILE" | awk '{print $1, $2, substr($0,
index($0,$4))}' | sed 's/ERROR //'
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano assignment6.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano sample.log

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./assignment6.sh
2024-05-25 14:23:00 Failed to connect to database
2024-05-25 14:35:01 Timeout occurred while processing
```

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.

```
file1= "old.txt"

file2= "new.txt"

sed '' "$file1" > "$file2"
```

output:

```
Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ nano Assignment7.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cat old.txt
Hello world

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cat new.txt

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ ./Assignment7.sh

Admin@LAPTOP-O3UEKIHS MINGW64 /d/Wipro/shell
$ cat new.txt
Hello world
```