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".

Program:

```
vi myfile.txt
vi search.sh
FILE="myfile.txt"
if [ -f "$FILE" ]; then
   echo "File exists"
else
   echo "File not found"
fi
chmod +x search.sh
./ search.sh
```

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.

Program:

```
vi number.sh
while true; do
   read -p "Enter a number (0 to stop): " number
   if [ "$number" -eq 0 ]; then
      echo "Exiting the script."
      break
   fi
odd_or_even() {
   if (( $1 % 2 == 0 )); then
```

```
        echo "The number $1 is even."
    else
        echo "The number $1 is odd."
    fi
}
odd_or_even $number
done
chmod +x number.sh
./ number.sh
```

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.

Program:

```
vi count.sh
count_lines() {
    local filename=$1
    if [[ -f $filename ]]; then
        local line_count=$(wc -l < "$filename")
        echo "The file '$filename' has $line_count lines."
    else
        echo "The file '$filename' was not found."
    fi
}
count_lines file1.txt
count_lines file2.txt
count_lines file3.txt
done
```

```
chmod +x count.sh
./count.sh
```

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").

Program:

```
vi create_files.sh
mkdir -p TestDir
cd TestDir
for i in {1..10}
do
  filename="File${i}.txt"
  echo $filename > $filename
done
echo "10 files have been created in the TestDir directory."
chmod +x create_files.sh
./ create_files.sh
```

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.

Program:

```
 vi error.sh
DEBUG=${DEBUG:-0}
debug() {
```

```
    if [ "$DEBUG" -eq 1 ]; then
        echo "DEBUG: $1"
    fi
}
create_files() {
    local dir=$1
    if [ -d "$dir" ]; then
        debug "The directory '$dir' already exists."
    else
        mkdir -p "$dir" 2>/dev/null
        if [ $? -ne 0 ]; then
            echo "Error: Unable to create directory '$dir'. Check permissions."
            exit 1
        fi
        debug "Created directory '$dir'."
    fi
    cd "$dir" || { echo "Error: Unable to change to directory '$dir'."; exit 1; }
    debug "Changed to directory '$dir'."

    for i in {1..5}
    do
        local filename="File${i}.txt"
              echo "$filename" > "$filename" 2>/dev/null
        if [ $? -ne 0 ]; then
            echo "Error: Unable to create file '$filename'. Check permissions."
            exit 1
        fi
```

```
        debug "Created file '$filename' with content '$filename'."
    done
    echo "10 files have been created in the '$dir' directory."
}
TARGET_DIR="TestDir"
create_files "$TARGET_DIR"
chmod +x error.sh
./ error.sh
```

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

Program:

```
vi log.sh
if [ "$#" -ne 1 ]; then
    echo "Usage: $0 log_file"
    exit 1
fi
log_file=$1
if [ ! -f "$log_file" ]; then
    echo "The file '$log_file' does not exist."
    exit 1
fi
grep "ERROR" "$log_file" | awk '{print $1, $2, $5}' | sed 's/\[ERROR\]//'
chmod +x log.sh
./log.sh
```

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.

Program:

```
vi replace.sh
if [ "$#" -ne 3 ]; then
    echo "Usage: $0 input_file old_text new_text"
    exit 1
fi
input_file=$1
old_text=$2
new_text=$3
output_file="output_${input_file}"

if [ ! -f "$input_file" ]; then
    echo "The file '$input_file' does not exist."
    exit 1
fi

sed "s/${old_text}/${new_text}/g" "$input_file" > "$output_file"
echo "Replaced all occurrences of '$old_text' with '$new_text' in '$input_file' "
chmod +x replace.sh
./ replace.sh
```