

Name : Tejaswini Anil Kamble

Batch Name: CPPE_Java Full Stack

Email: teju000kamble@gmail.com

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".

Ans : code

```
filename="Myfile1.txt"
if [ -e "$filename" ]; then
    echo "File exists"
else
    echo "File not found"
fi
```

Output :

```
main.bash  Myfile1.txt  ⋮
1  #!/bin/bash
2
3  filename="Myfile1.txt"
4
5
6  if [ -e "$filename" ]; then
7      echo "File exists"
8  else
9      echo "File not found"
10 fi

File exists

...Program finished with exit code 0
Press ENTER to exit console. □
```

Assignment 2

Q. 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.

Ans: Code

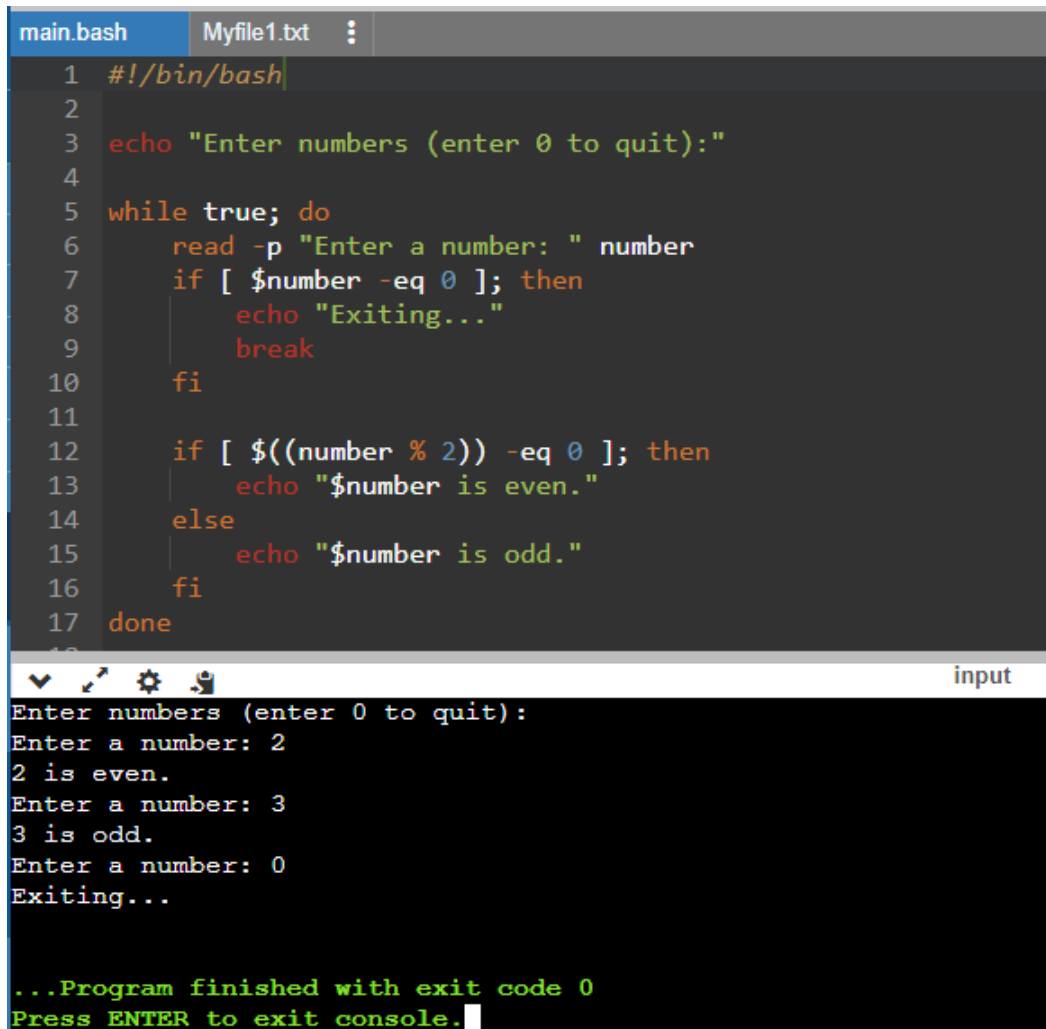
```
#!/bin/bash
```

```
echo "Enter numbers (enter 0 to quit):"
```

```
while true; do
    read -p "Enter a number: " 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
```

Output:



The screenshot shows a terminal window with a dark background. The top bar has tabs for 'main.bash' and 'Myfile1.txt'. The script content is as follows:

```
1  #!/bin/bash
2
3  echo "Enter numbers (enter 0 to quit):"
4
5  while true; do
6      read -p "Enter a number: " number
7      if [ $number -eq 0 ]; then
8          echo "Exiting..."
9          break
10     fi
11
12     if [ $((number % 2)) -eq 0 ]; then
13         echo "$number is even."
14     else
15         echo "$number is odd."
16     fi
17 done
```

The output of the script is shown below the script content:

```
Enter numbers (enter 0 to quit):
Enter a number: 2
2 is even.
Enter a number: 3
3 is odd.
Enter a number: 0
Exiting...

...Program finished with exit code 0
Press ENTER to exit console.
```

Assignment 3

Q. 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.

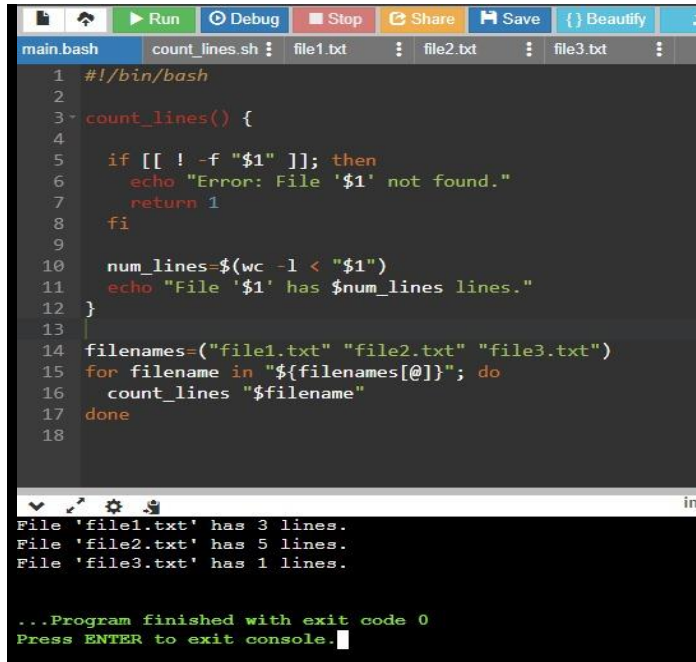
Ans: code

```
#!/bin/bassh

file1.txt
file2.txt
count_lines() {
if [ -f "$1" ]; then
fi
echo "Error: File '$1' not found."
return 1
file3.bd
}
num_lines=$(wc -l < "$1")
echo "File '$1' has $num_lines lines."

filenames=("file1.txt" "file2.txt" "file3.txt")
for filename in "${filenames[@]}; do
count_lines "$filename
done
```

Output:



```
1  #!/bin/bash
2
3  count_lines() {
4
5      if [[ ! -f "$1" ]]; then
6          echo "Error: File '$1' not found."
7          return 1
8      fi
9
10     num_lines=$(wc -l < "$1")
11     echo "File '$1' has $num_lines lines."
12 }
13
14 filenames=("file1.txt" "file2.txt" "file3.txt")
15 for filename in "${filenames[@]}; do
16     count_lines "$filename"
17 done
18
```

File 'file1.txt' has 3 lines.
File 'file2.txt' has 5 lines.
File 'file3.txt' has 1 lines.

...Program finished with exit code 0
Press ENTER to exit console.

Assignment 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").

Ans : Code

```
mkdir -p TestDir
```

```
for ((i = 1; i <= 10; i++)); do
```

```
    filename="DemoFile${i}.txt"
```

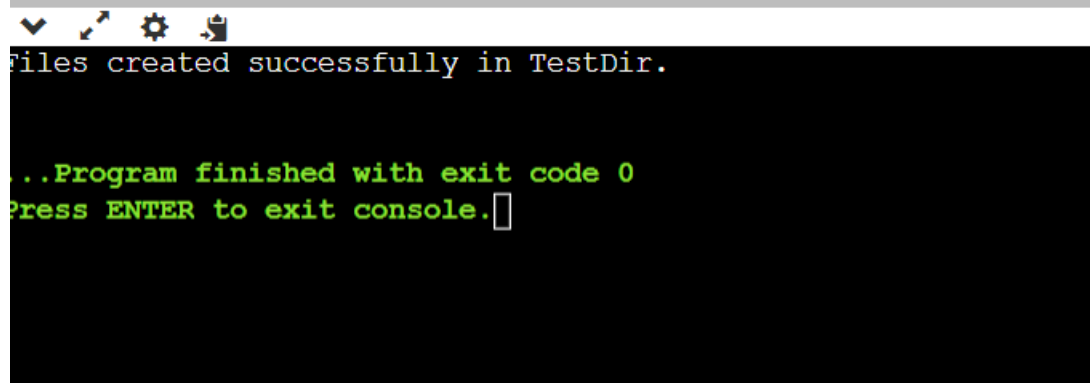
```
    echo "$filename" > "TestDir/$filename"
```

```
done
```

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

Output:

```
62
63 mkdir -p TestDir
64
65
66 for ((i = 1; i <= 10; i++)); do
67     filename="DemoFile${i}.txt"
68     echo "$filename" > "TestDir/$filename"
69 done
70
71 echo "Files created successfully in TestDir."
72
73
74
75
76
```



Assignment 5

Q. Modify the script to handle errors, such as the directory already existing or lacking permissions to create files.

Ans : Code

```
create_directory() {
    local dir_name="$1"
    if mkdir "$dir_name" 2>/dev/null; then
        echo "Directory '$dir_name' created successfully."
    else
```

```
if [ -d "$dir_name" ]; then
    echo "Directory '$dir_name' already exists."
else
    echo "Error: Could not create directory '$dir_name'."
fi
fi
}

create_files() {
    local dir_name="$1"
    local num_files="$2"
    for i in $(seq 1 "$num_files"); do
        local file_name="File$i.txt"
        local file_path="$dir_name/$file_name"
        if echo "$file_name" > "$file_path"; then
            echo "File '$file_name' created successfully in '$dir_name'."
        else
            echo "Error: Could not create file '$file_name' in directory '$dir_name'."
        fi
    done
}

main() {
    local dir_name="TestDir"
```

```
local num_files=10

create_directory "$dir_name"

create_files "$dir_name" "$num_files"

}main
```

Output:

```
19 |
20 - create_files() {
21     local dir_name="$1"
22     local num_files="$2"
23     for i in $(seq 1 "$num_files"); do
24         local file_name="File$i.txt"
25         local file_path="$dir_name/$file_name"
26         if echo "$file_name" > "$file_path"; then
27             echo "File '$file_name' created successfully in '$dir_name'."
28         else
29             echo "Error: Could not create file '$file_name' in directory '$dir_name'."
30         fi
31     done
32 }
33
34
35 - main() {
input
Directory 'TestDir' created successfully.
File 'File1.txt' created successfully in 'TestDir'.
File 'File2.txt' created successfully in 'TestDir'.
File 'File3.txt' created successfully in 'TestDir'.
File 'File4.txt' created successfully in 'TestDir'.
File 'File5.txt' created successfully in 'TestDir'.
File 'File6.txt' created successfully in 'TestDir'.
File 'File7.txt' created successfully in 'TestDir'.
File 'File8.txt' created successfully in 'TestDir'.
File 'File9.txt' created successfully in 'TestDir'.
File 'File10.txt' created successfully in 'TestDir'.

...Program finished with exit code 0
Press ENTER to exit console.[]
```


Assignment 6

Q.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.

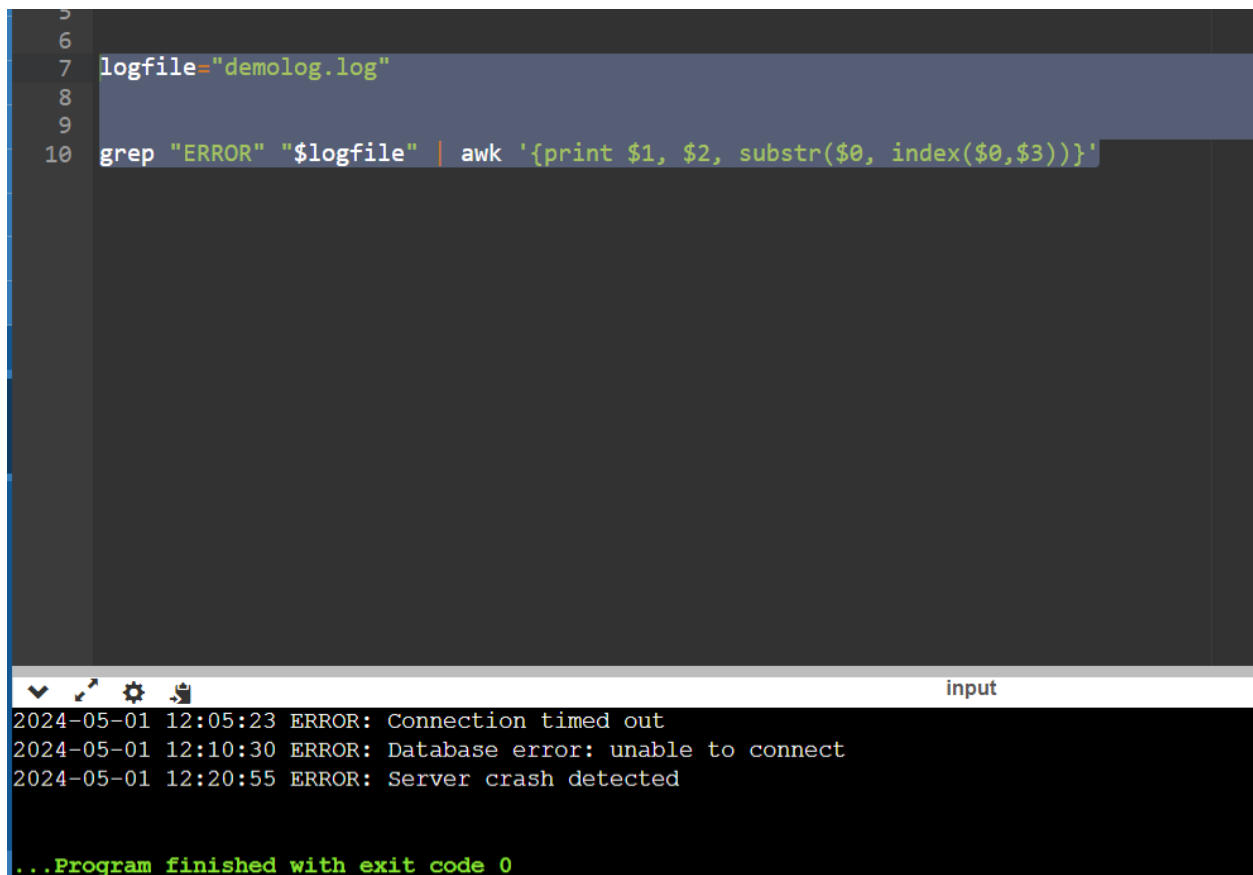
Data Processing with sed

Ans : Code

```
logfile="demolog.log"
```

```
grep "ERROR" "$logfile" | awk '{print $1, $2, substr($0, index($0,$3))}'
```

Output:



```
5
6
7 logfile="demolog.log"
8
9
10 grep "ERROR" "$logfile" | awk '{print $1, $2, substr($0, index($0,$3))}'

input
2024-05-01 12:05:23 ERROR: Connection timed out
2024-05-01 12:10:30 ERROR: Database error: unable to connect
2024-05-01 12:20:55 ERROR: Server crash detected
...Program finished with exit code 0
```

Assignment 7

Q. 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.

Ans : Code

```
input_file="file1.txt"
```

```
old_text="linux"
```

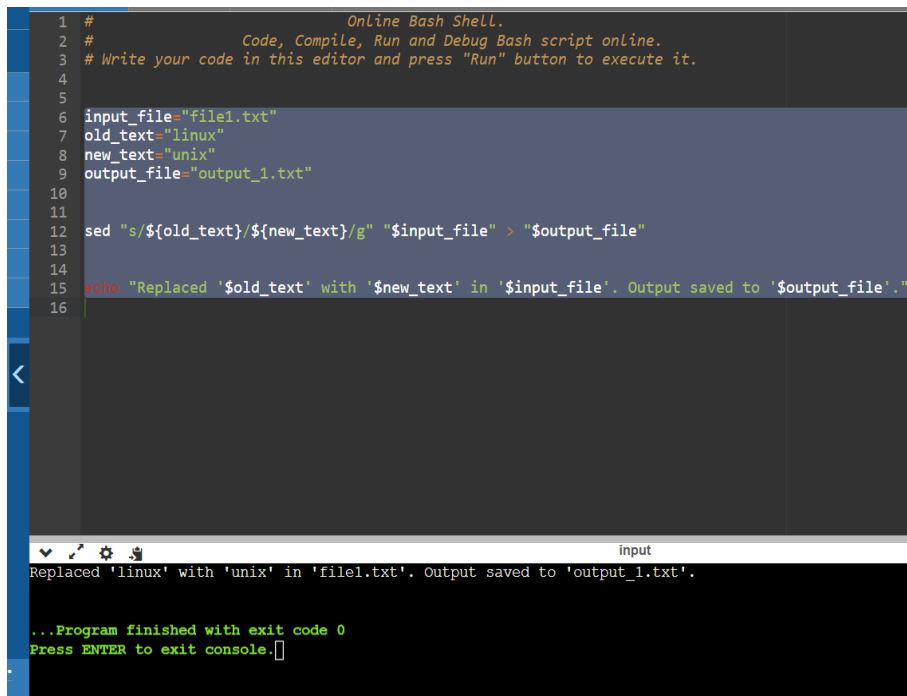
```
new_text="unix"
```

```
output_file="output_1.txt"
```

```
sed "s/${old_text}/${new_text}/g" "$input_file" > "$output_file"
```

```
echo "Replaced '$old_text' with '$new_text' in '$input_file'. Output saved to '$output_file'."
```

Output :



```
1 # Online Bash Shell.
2 # Code, Compile, Run and Debug Bash script online.
3 # Write your code in this editor and press "Run" button to execute it.
4
5
6 input_file="file1.txt"
7 old_text="linux"
8 new_text="unix"
9 output_file="output_1.txt"
10
11
12 sed "s/${old_text}/${new_text}/g" "$input_file" > "$output_file"
13
14
15 echo "Replaced '$old_text' with '$new_text' in '$input_file'. Output saved to '$output_file'."
16
```

Input

```
Replaced 'linux' with 'unix' in 'file1.txt'. Output saved to 'output_1.txt'.
```

...Program finished with exit code 0
Press ENTER to exit console.

