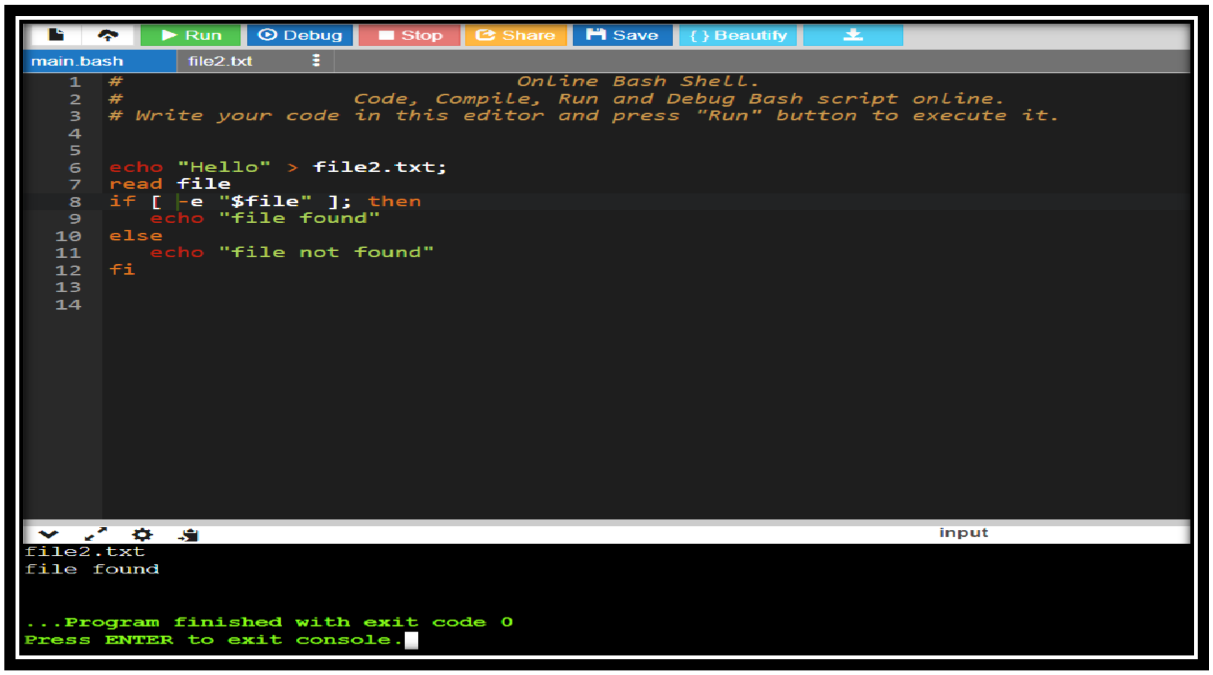
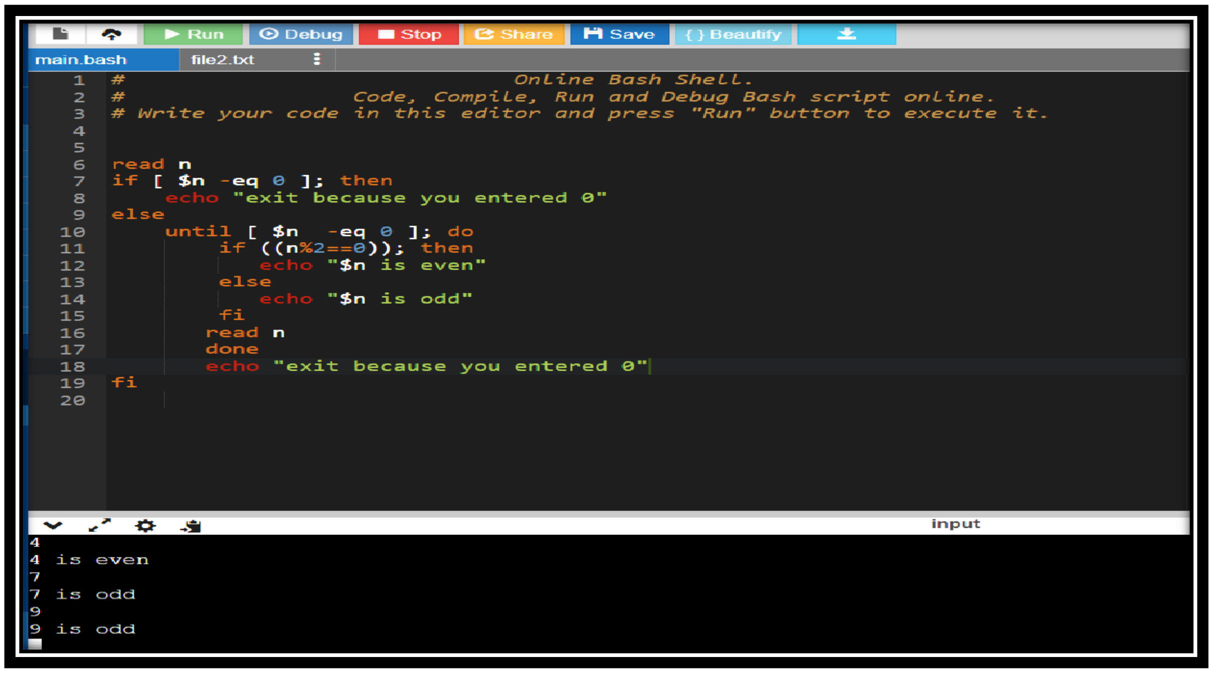
1) Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".



```bash
#                                     Online Bash Shell.
#                       Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.


echo "Hello" > file2.txt;
read file
if [ -e "$file" ]; then
    echo "file found"
else
    echo "file not found"
fi
```

```
file2.txt
file found


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

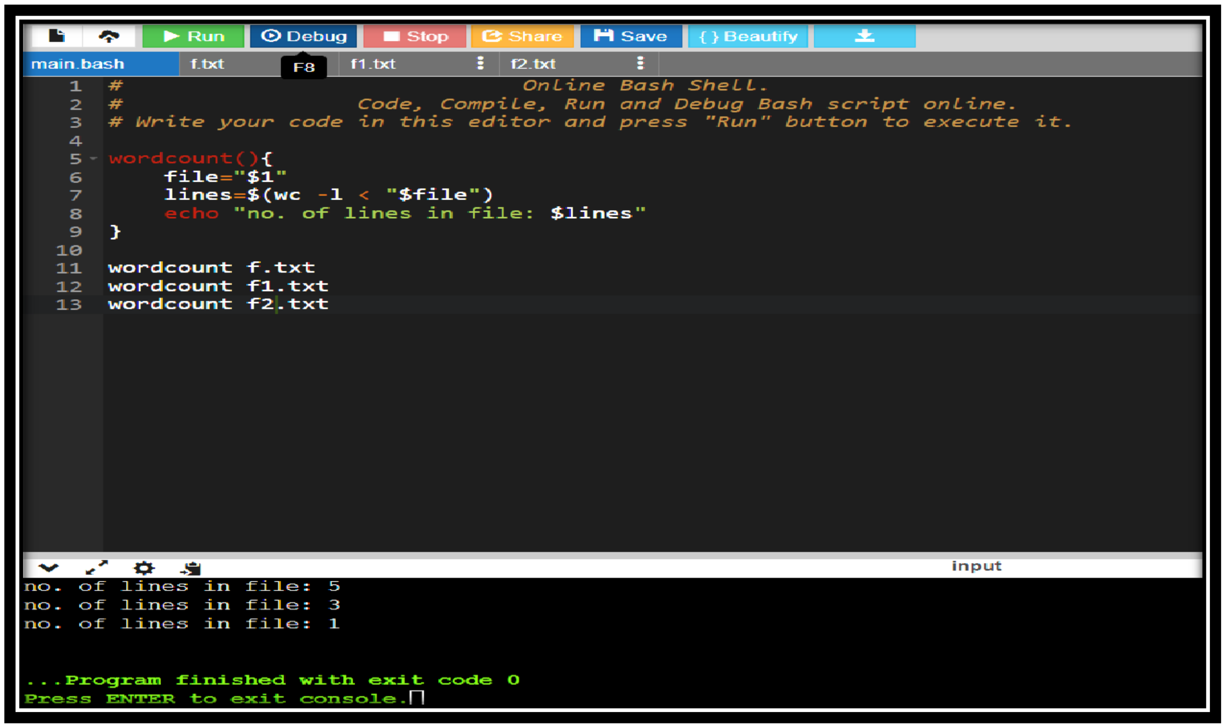
2) Write a script that reads numbers from the user until they enter '0'. The script should also print whether each number is odd or even.



```bash
#                                     Online Bash Shell.
#                       Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.


read n
if [ $n -eq 0 ]; then
    echo "exit because you entered 0"
else
    until [ $n  -eq 0 ]; do
        if ((n%2==0)); then
           echo "$n is even"
        else
           echo "$n is odd"
        fi
       read n
       done
       echo "exit because you entered 0"
fi
```

```
4
4 is even
7
7 is odd
9
9 is odd
```

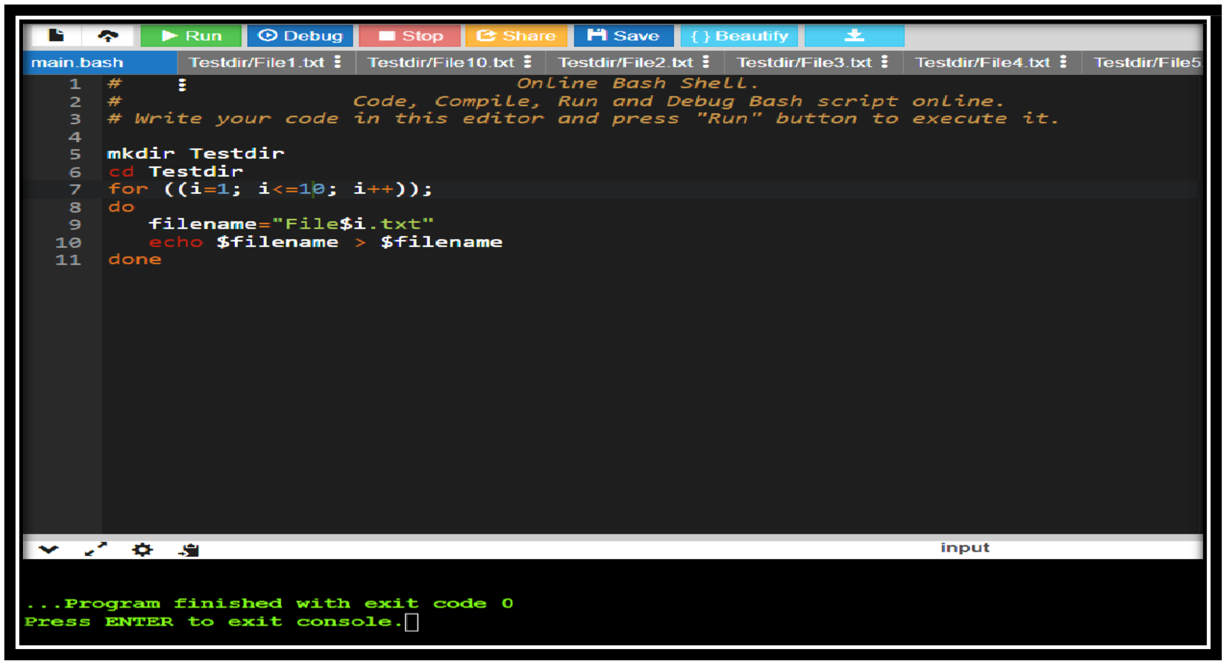
3) Create a function that takes a filename as an argument and prints the number of lines inthe file. Call this function from your script with different filenames.



```
#                            Online Bash Shell.
#                Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.

wordcount(){
    file="$1"
    lines=$(wc -l < "$file")
    echo "no. of lines in file: $lines"
}

wordcount f.txt
wordcount f1.txt
wordcount f2.txt
```

```
no. of lines in file: 5
no. of lines in file: 3
no. of lines in file: 1


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

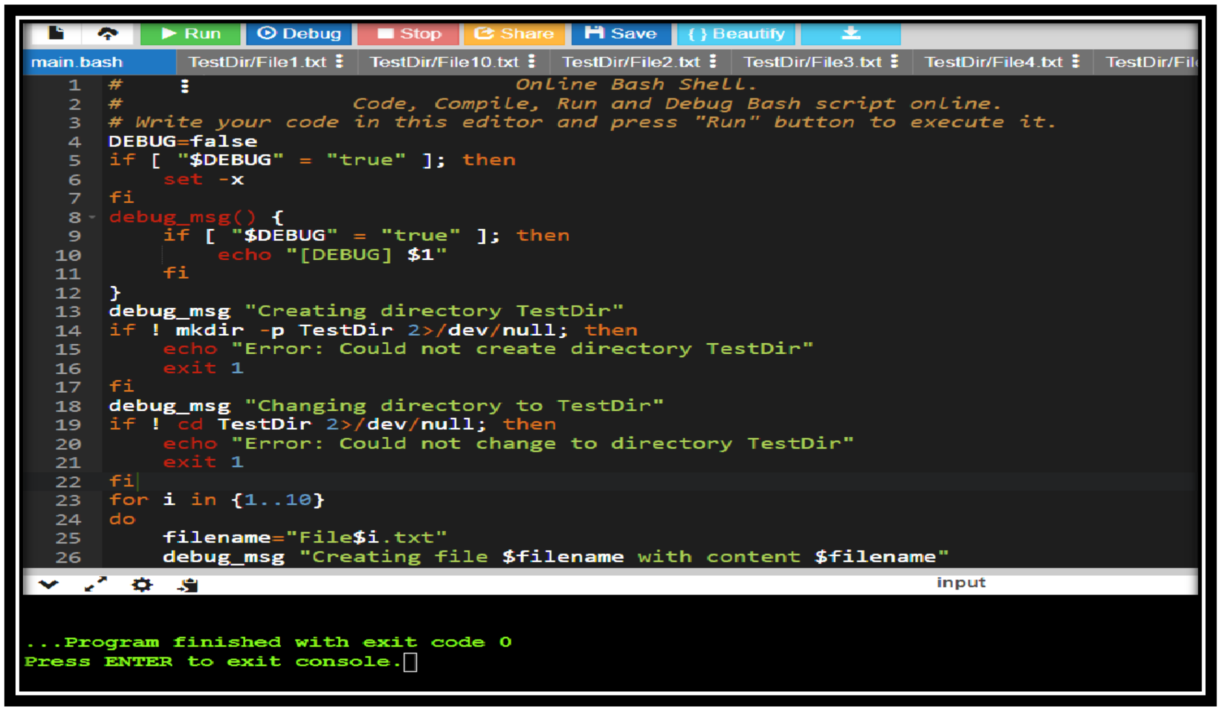
4) Write a script that creates a directory named TestDir and inside it, creates ten files namedFile1.txt, File2.txt, ... File10.txt. Each file should contain its filename as its content (e.g., File1.txt contains "File1.txt").



```
#                            Online Bash Shell.
#                Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.

mkdir Testdir
cd Testdir
for ((i=1; i<=10; i++));
do
   filename="File$i.txt"
   echo $filename > $filename
done
```

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

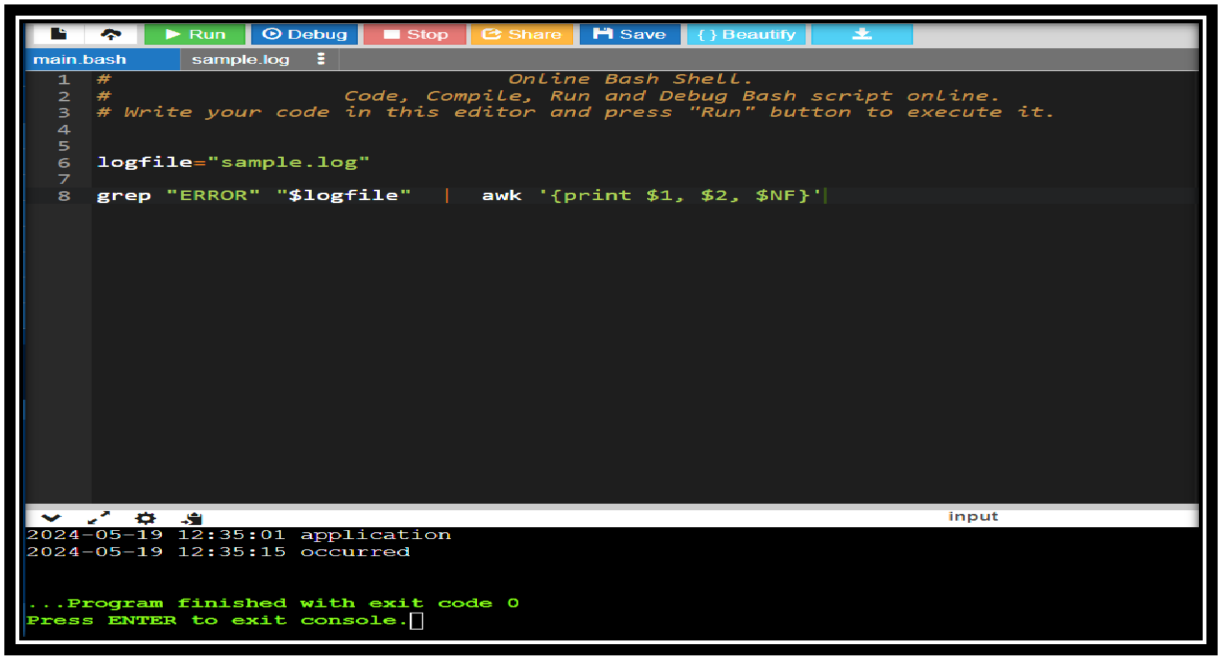
5) Modify the script to handle errors, such as the directory already existing or lacking permissions to create files. Add a debugging mode that prints additional information when enabled.



```
#     :                         Online Bash Shell.
#                  Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.
DEBUG=false
if [ "$DEBUG" = "true" ]; then
    set -x
fi
debug_msg() {
    if [ "$DEBUG" = "true" ]; then
        echo "[DEBUG] $1"
    fi
}
debug_msg "Creating directory TestDir"
if ! mkdir -p TestDir 2>/dev/null; then
    echo "Error: Could not create directory TestDir"
    exit 1
fi
debug_msg "Changing directory to TestDir"
if ! cd TestDir 2>/dev/null; then
    echo "Error: Could not change to directory TestDir"
    exit 1
fi
for i in {1..10}
do
    filename="File$i.txt"
    debug_msg "Creating file $filename with content $filename"
```

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

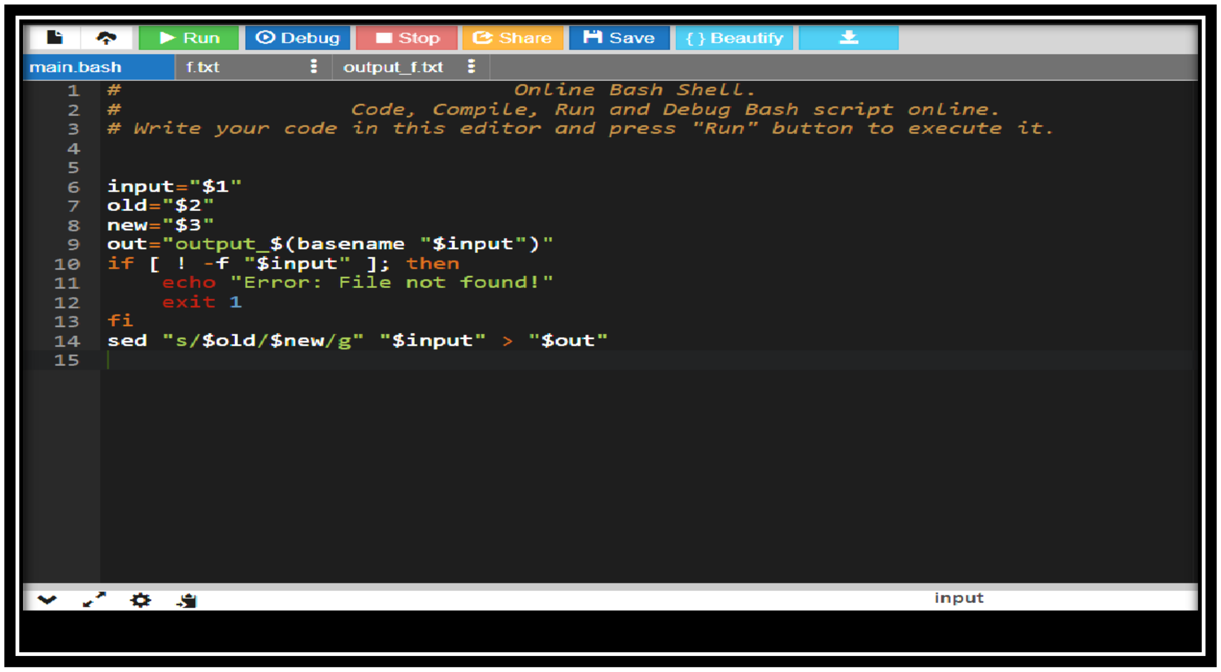
6) Given a sample log file, write a script using grep to extract all lines containing "ERROR".Use awk to print the date, time, and error message of each extracted line.



```
#                               Online Bash Shell.
#                  Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.


logfile="sample.log"

grep "ERROR" "$logfile"  |  awk '{print $1, $2, $NF}'
```

```
2024-05-19 12:35:01 application
2024-05-19 12:35:15 occurred


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

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.



```bash
#                                 Online Bash Shell.
#                    Code, Compile, Run and Debug Bash script online.
# Write your code in this editor and press "Run" button to execute it.


input="$1"
old="$2"
new="$3"
out="output_$(basename "$input")"
if [ ! -f "$input" ]; then
    echo "Error: File not found!"
    exit 1
fi
sed "s/$old/$new/g" "$input" > "$out"
```