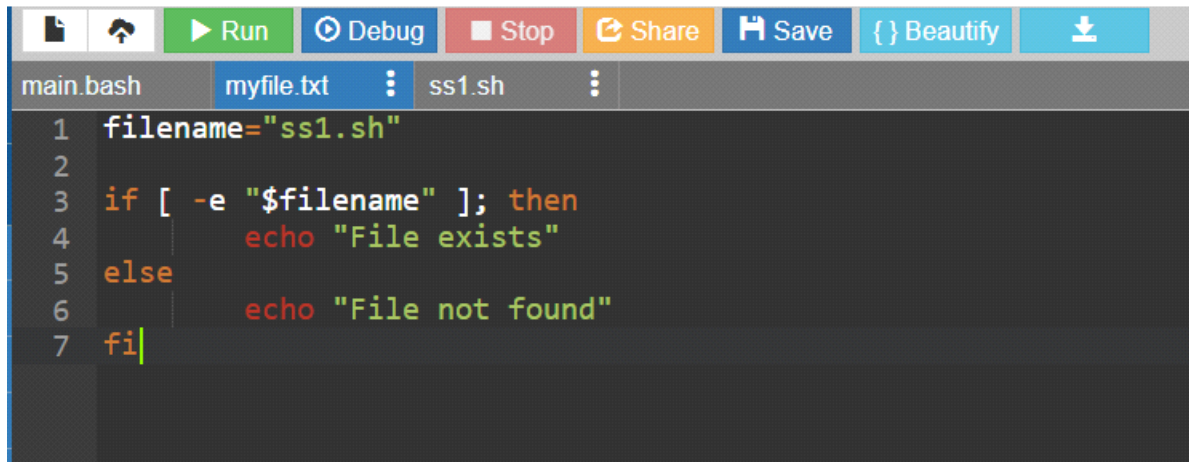
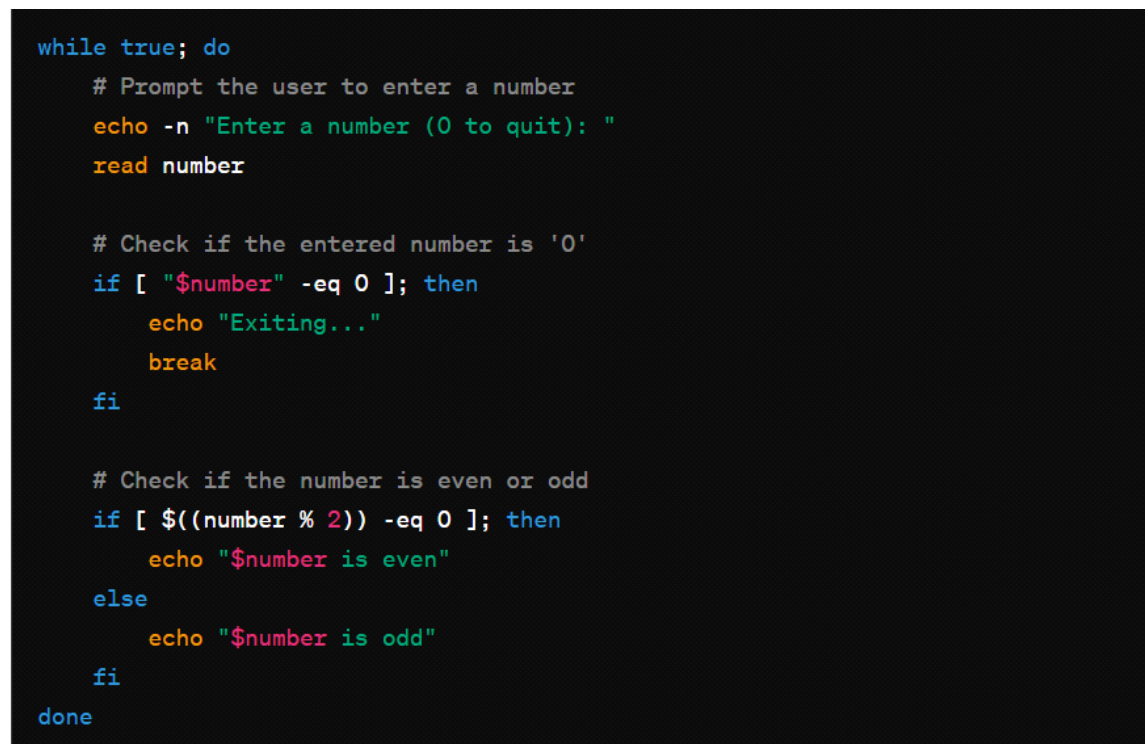


Assignment 1: Ensure the script checks if a specific file (e.g., myfile.txt) exists in the current directory. If it exists, print "File exists", otherwise print "File not found".

A screenshot of a code editor interface. At the top, there is a toolbar with icons for file operations and buttons labeled 'Run', 'Debug', 'Stop', 'Share', 'Save', 'Beautify', and a download icon. Below the toolbar, a tab bar shows three tabs: 'main.bash', 'myfile.txt', and 'ss1.sh'. The 'ss1.sh' tab is active, displaying a shell script with the following content:

```
1 filename="ss1.sh"
2
3 if [ -e "$filename" ]; then
4     echo "File exists"
5 else
6     echo "File not found"
7 fi
```

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

A screenshot of a code editor showing a shell script. The script is as follows:

```
while true; do
    # Prompt the user to enter a number
    echo -n "Enter a number (0 to quit): "
    read number

    # Check if the entered number is '0'
    if [ "$number" -eq 0 ]; then
        echo "Exiting..."
        break
    fi

    # Check if the number is even or odd
    if [ $((number % 2)) -eq 0 ]; then
        echo "$number is even"
    else
        echo "$number is odd"
    fi
done
```

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.

```

count_lines_in_file() {
    local filename="$1"
    if [ -f "$filename" ]; then
        local line_count=$(wc -l < "$filename")
        echo "$line_count"
    else
        echo "File not found"
        return 1
    fi
}

# Main script
for fname in "$@"; do
    line_count=$(count_lines_in_file "$fname")
    if [ $? -eq 0 ]; then
        echo "The file '$fname' has $line_count lines."
    else
        echo "The file '$fname' was not found."
    fi
done

```

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").

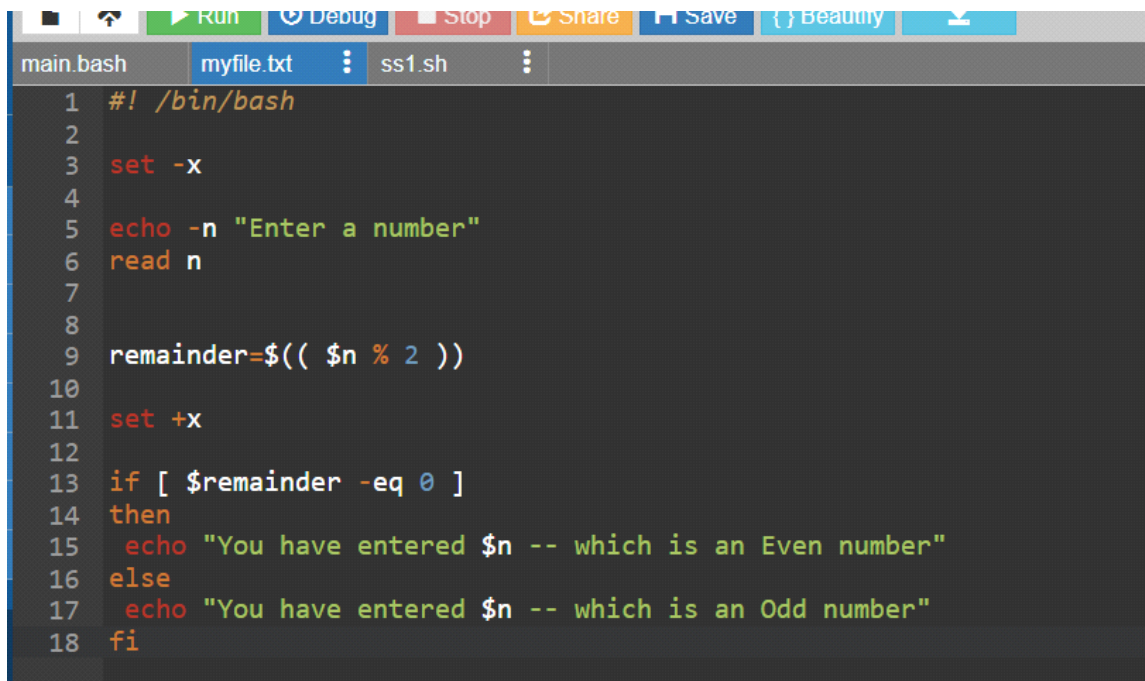
```

rps@rps-virtual-machine:~$ mkdir testdir
rps@rps-virtual-machine:~$ cd testdir
rps@rps-virtual-machine:~/testdir$ echo "File1.txt" > File1.txt
rps@rps-virtual-machine:~/testdir$ echo "File2.txt" > File2.txt
rps@rps-virtual-machine:~/testdir$ echo "File3.txt" > File3.txt
rps@rps-virtual-machine:~/testdir$ echo "File4.txt" > File4.txt
rps@rps-virtual-machine:~/testdir$ echo "File5.txt" > File5.txt
rps@rps-virtual-machine:~/testdir$ echo "File6.txt" > File6.txt
rps@rps-virtual-machine:~/testdir$ echo "File7.txt" > File7.txt
rps@rps-virtual-machine:~/testdir$ echo "File8.txt" > File8.txt
rps@rps-virtual-machine:~/testdir$ echo "File9.txt" > File9.txt
rps@rps-virtual-machine:~/testdir$ echo "File10.txt" > File10.txt
rps@rps-virtual-machine:~/testdir$ ls
File10.txt  File2.txt  File4.txt  File6.txt  File8.txt
File1.txt   File3.txt  File5.txt  File7.txt  File9.txt
rps@rps-virtual-machine:~/testdir$ cat file1.txt
cat: file1.txt: No such file or directory
rps@rps-virtual-machine:~/testdir$ cat File1.txt
File1.txt
rps@rps-virtual-machine:~/testdir$

```

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.



```

1  #!/bin/bash
2
3  set -x
4
5  echo -n "Enter a number"
6  read n
7
8
9  remainder=$(( $n % 2 ))
10
11 set +x
12
13 if [ $remainder -eq 0 ]
14 then
15     echo "You have entered $n -- which is an Even number"
16 else
17     echo "You have entered $n -- which is an Odd number"
18 fi

```

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

```
bash myfile.txt ss1.sh sample.log
1 #!/bin/bash
2
3 # Define the log file
4 LOGFILE="sample.log"
5
6 # Use grep to extract lines containing "ERROR"
7 grep "ERROR" "$LOGFILE" | awk '
8 {
9     # Assuming the log format is: YYYY-MM-DD HH:MM:SS [ERROR] Error message
10     date=$1
11     time=$2
12     # Combine the rest of the line for the error message
13     $1=$2=$3=""
14     error_message=$0
15
16     # Print date, time and error message
17     print date, time, error_message
18 }' | sed 's/^\\s*//;s/\\s*$//'
```

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.

```
rps@rps-virtual-machine:~/testdir$ vim file1.txt
rps@rps-virtual-machine:~/testdir$ cat file1.txt

hello world java
rps@rps-virtual-machine:~/testdir$ sed -i 's/hello world java/amit kumar/' file1.txt
rps@rps-virtual-machine:~/testdir$ cat file1.txt

amit kumar
rps@rps-virtual-machine:~/testdir$
```