BCSE – 5th Semester – 2025-26 Assignment – I [CO1]

Operating Systems Lab

Name : Tathagata Sur

Roll: 002310501030 (A1 Group)

QUESTION 1:

Write a shell script that asks for the values of 2 variables: userv1 and userv2, from the user and takes in any type (real/integer/character). Try to perform the following operations on the variables: (a) add, (b) multiply (c) subtract (d) division (e) printing the variables in reverse order [If userv1 is 1234/abcd, then output should be 4321/dcba. Print appropriate error message/s if the operation cannot be performed, otherwise print the result: "the sum of 'userv1' and 'userv2' is ....". Execute the shell script as many times the user wants. Show all possible combinations of the types of variables and result/s of the corresponding operation/s.

ANSWER :

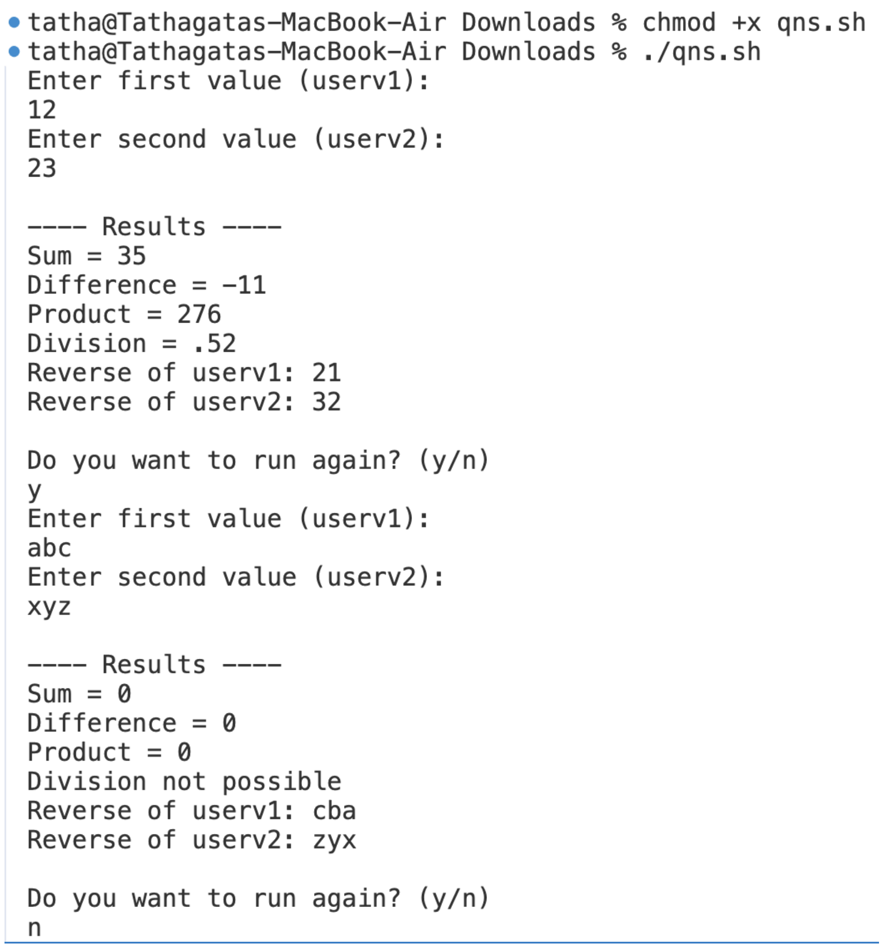
```
#!/bin/bash

while true

do

    # Getting user input for two variables

    echo "Enter first value (userv1): "

    read userv1

    echo "Enter second value (userv2): "

    read userv2

    echo

    echo "---- Results ----"
```

```
# Arithmetic operations using bc(Basic Calculator) for calculation
sum=$(echo "$userv1 + $userv2" | bc 2>/dev/null)
if [ -n "$sum" ]; then
    echo "Sum = $sum"
else
    echo "Addition not possible"
fi

diff=$(echo "$userv1 - $userv2" | bc 2>/dev/null)
if [ -n "$diff" ]; then
    echo "Difference = $diff"
else
    echo "Subtraction not possible"
fi

prod=$(echo "$userv1 * $userv2" | bc 2>/dev/null)
if [ -n "$prod" ]; then
    echo "Product = $prod"
else
    echo "Multiplication not possible"
fi

# Checking for division by zero
if [ "$userv2" = "0" ] || [ "$userv2" = "0.0" ]; then
    echo "Division not possible (divide by zero)"
else
```

```
    div=$(echo "scale=2; $userv1 / $userv2" | bc 2>/dev/null)
    if [ -n "$div" ]; then
        echo "Division = $div"
    else
        echo "Division not possible"
    fi
fi

# Function to reverse string/number
reverse_string() {
    str="$1"
    len=${#str}
    revstr=""
    for (( i=$len-1; i>=0; i-- )); do
        revstr="$revstr${str:$i:1}"
    done
    echo "$revstr"
}
echo "Reverse of userv1: $(reverse_string "$userv1")"
echo "Reverse of userv2: $(reverse_string "$userv2")"
echo

# Ask user to continue or exit
echo "Do you want to run again? (y/n)"
read ans
[ "$ans" != "y" ] && break
```

done

Example Output:



```
tatha@Tathagatas-MacBook-Air Downloads % chmod +x qns.sh
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh
Enter first value (userv1):
12
Enter second value (userv2):
23

---- Results ----
Sum = 35
Difference = -11
Product = 276
Division = .52
Reverse of userv1: 21
Reverse of userv2: 32

Do you want to run again? (y/n)
y
Enter first value (userv1):
abc
Enter second value (userv2):
xyz

---- Results ----
Sum = 0
Difference = 0
Product = 0
Division not possible
Reverse of userv1: cba
Reverse of userv2: zyx

Do you want to run again? (y/n)
n
```

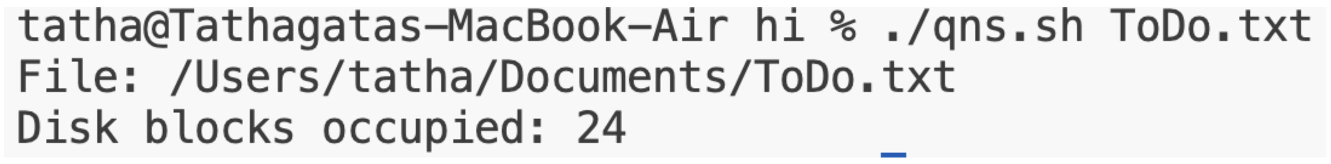
QUESTION 2:

Write a shell script that counts the number of disk blocks occupied by a file (the file name is passed as an argument and the file may be anywhere in the system, but not in the current directory).

ANSWER :

```
#!/bin/bash
# Checking if filename argument provided
if [ $# -ne 1 ]; then
  echo "Usage: $0 <filename>"
  exit 1
fi
filename=$1

# Finding file from home directory, excluding current directory
filepath=$(find ~ -name "$filename" -type f ! -path "$(pwd)/*" 2>/dev/null | head
-1)
if [ -z "$filepath" ]; then
  echo "File '$filename' not found outside current directory."
  exit 1
fi


# Counting disk blocks using stat command
blocks=$(stat -c %b "$filepath" 2>/dev/null || stat -f %b "$filepath"
2>/dev/null)
echo "File: $filepath"
echo "Disk blocks occupied: $blocks"
```

Example Output:



```
tatha@Tathagatas-MacBook-Air hi % ./qns.sh ToDo.txt
File: /Users/tatha/Documents/ToDo.txt
Disk blocks occupied: 24
```

QUESTION 3:

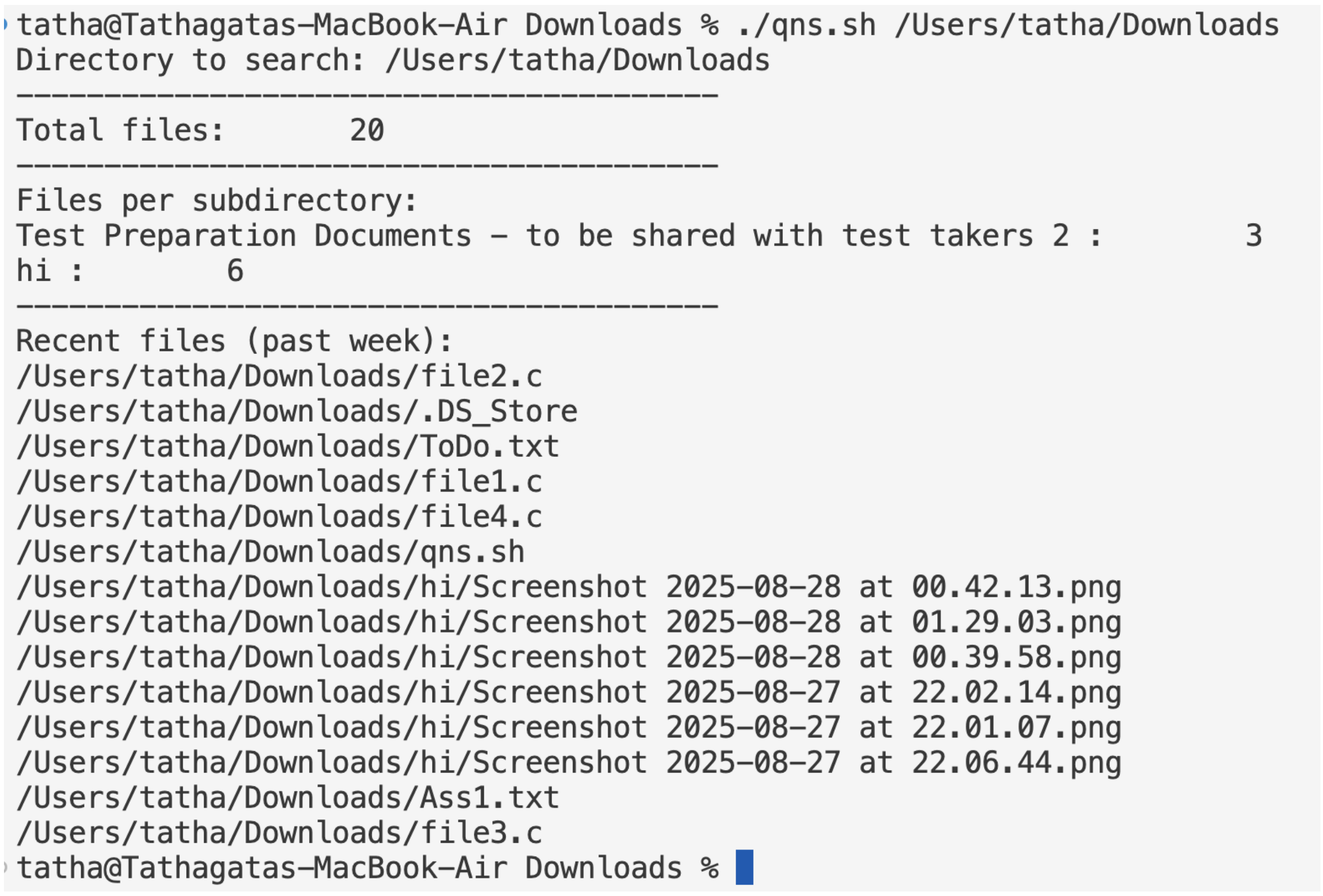
Write a shell script that counts and prints the total number of files in a particular directory and in all its sub-directories. Also (i) print total no. of files (not directories) in each subdirectory with the name of the subdirectory, (ii) print only those file names that have been created within the past week.

ANSWER :

```
#!/bin/bash
# Check directory argument
```

```
if [ $# -eq 0 ]; then
   echo "Usage: $0 <directory_path>"
   exit 1
fi
DIR=$1
if [ ! -d "$DIR" ]; then
   echo "Error: $DIR is not a directory"
   exit 1
fi

echo "Directory to search: $DIR"
echo "----------------------------------------"

# Counting all files recursively using find and wc
total_files=$(find "$DIR" -type f | wc -l)
echo "Total files: $total_files"
echo "----------------------------------------"

# Counting files per subdirectory using simple loop
echo "Files per subdirectory:"
for subdir in "$DIR"/*/; do
   if [ -d "$subdir" ]; then
       count=$(find "$subdir" -maxdepth 1 -type f | wc -l)
       echo "$(basename "$subdir") : $count"
   fi
done
echo "----------------------------------------"

# Files modified in last 7 days
echo "Recent files (past week):"
find "$DIR" -type f -mtime -7
```

Example Output: (PTO)



```
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh /Users/tatha/Downloads
Directory to search: /Users/tatha/Downloads
----------------------------------------
Total files:       20
----------------------------------------
Files per subdirectory:
Test Preparation Documents - to be shared with test takers 2 :        3
hi :        6
----------------------------------------
Recent files (past week):
/Users/tatha/Downloads/file2.c
/Users/tatha/Downloads/.DS_Store
/Users/tatha/Downloads/ToDo.txt
/Users/tatha/Downloads/file1.c
/Users/tatha/Downloads/file4.c
/Users/tatha/Downloads/qns.sh
/Users/tatha/Downloads/hi/Screenshot 2025-08-28 at 00.42.13.png
/Users/tatha/Downloads/hi/Screenshot 2025-08-28 at 01.29.03.png
/Users/tatha/Downloads/hi/Screenshot 2025-08-28 at 00.39.58.png
/Users/tatha/Downloads/hi/Screenshot 2025-08-27 at 22.02.14.png
/Users/tatha/Downloads/hi/Screenshot 2025-08-27 at 22.01.07.png
/Users/tatha/Downloads/hi/Screenshot 2025-08-27 at 22.06.44.png
/Users/tatha/Downloads/Ass1.txt
/Users/tatha/Downloads/file3.c
tatha@Tathagatas-MacBook-Air Downloads %
```

QUESTION 4:

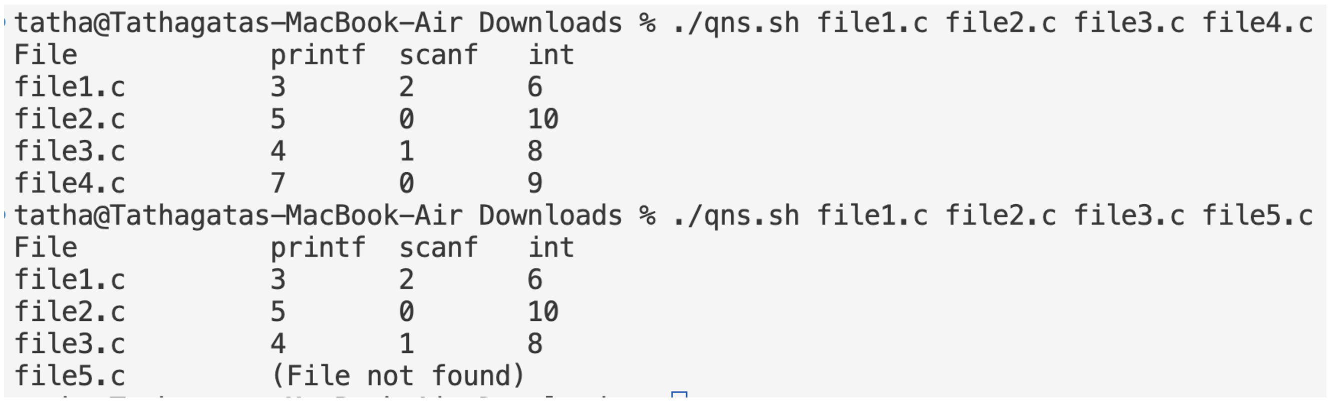
Write a shell script that takes 4 file names (C programs) as command line arguments and prints the frequency of the occurrences of the following 3 strings "printf", "scanf", "int" in each file. The output (in tabular format) should clearly denote the frequency of the occurrences of each string in each file.

ANSWER :

```
#!/bin/bash
# Check for 4 file arguments
[ $# -ne 4 ] && { echo "Usage: $0 file1.c file2.c file3.c file4.c"; exit 1; }

# Printing header
echo -e "File\t\tprintf\tscanf\tint"
```

```
# Count keywords in each file
for file in "$@"; do
  if [ -f "$file" ]; then
    printf "%-12s\t%d\t%d\t%d\n" "$file" \
      $(grep -o "printf" "$file" | wc -l) \
      $(grep -o "scanf" "$file" | wc -l) \
      $(grep -o "int" "$file" | wc -l)
  else
    printf "%-12s\t(File not found)\n" "$file"
  fi
done
```

Example Output:



```
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh file1.c file2.c file3.c file4.c
File            printf  scanf   int
file1.c         3       2       6
file2.c         5       0       10
file3.c         4       1       8
file4.c         7       0       9
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh file1.c file2.c file3.c file5.c
File            printf  scanf   int
file1.c         3       2       6
file2.c         5       0       10
file3.c         4       1       8
file5.c         (File not found)
```

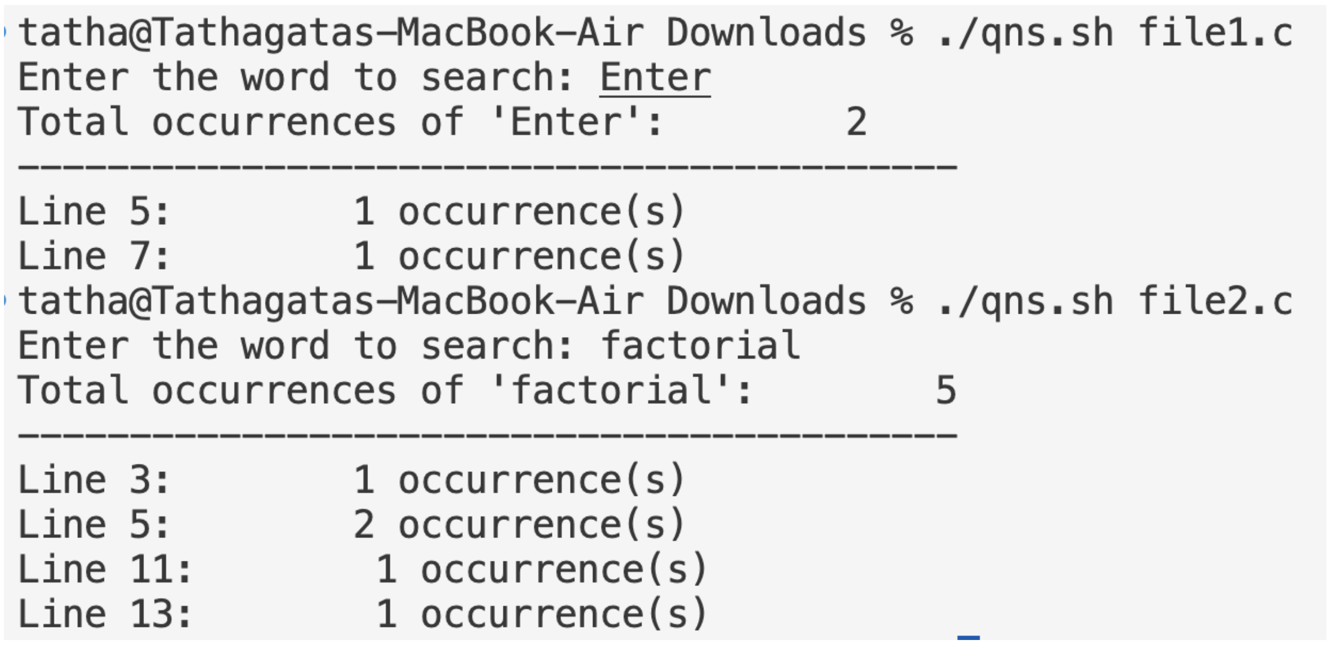
QUESTION 5:

Write a shell script that accepts a filename as input. The program then asks for a string of characters (that is, any word) to be provided by the user. The file will be searched to find whether it contains the given word. (i) If the file contains the given word, the program will display the total number of occurrences of the word. (ii) The program is also required to display the line number/s in which the word has occurred as well as the frequency of the word in that line. (Note: the word may occur multiple times in a given line). (iii) If the file does not contain the word, an appropriate message will be displayed. Ignore partial match, but the match has to be case sensitive.

ANSWER :

```bash
#!/bin/bash
# Checking arguments and file existence
[ $# -ne 1 ] && { echo "Usage: $0 filename"; exit 1; }
[ ! -f "$1" ] && { echo "File not found!"; exit 1; }

# Get search word from user
read -p "Enter the word to search: " word

# Count total occurrences
total=$(grep -ow "$word" "$1" | wc -l)
[ $total -eq 0 ] && { echo "The word '$word' was not found in the file."; exit 0; }

echo "Total occurrences of '$word': $total"
echo "----------------------------------------"

# Show line numbers and frequency per line
grep -nw "$word" "$1" | while IFS=: read -r lineno line; do
   count=$(echo "$line" | grep -ow "$word" | wc -l)
   echo "Line $lineno: $count occurrence(s)"
done
```

Example Output:



```
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh file1.c
Enter the word to search: Enter
Total occurrences of 'Enter':        2
------------------------------------------
Line 5:        1 occurrence(s)
Line 7:        1 occurrence(s)
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh file2.c
Enter the word to search: factorial
Total occurrences of 'factorial':        5
------------------------------------------
Line 3:        1 occurrence(s)
Line 5:        2 occurrence(s)
Line 11:        1 occurrence(s)
Line 13:        1 occurrence(s)
```

QUESTION 6:

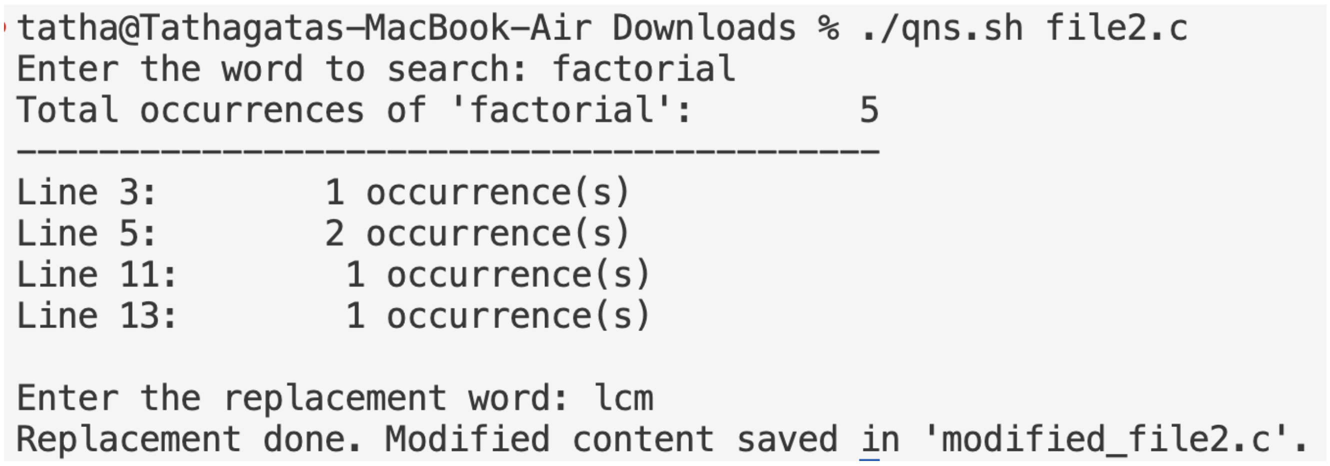
Extend the shell script written in question (5) to perform the following task: User is asked to enter another word. (i) The first word (entered in question (5)) to be replaced by the second word, if a match occurs, (ii) Ignore replacing Partial matches, but show that (iii) partial matches do exist, and (iv) match exists, if case sensitivity is ignored.

ANSWER :

```
#!/bin/bash
# Checking arguments and file existence
[ $# -ne 1 ] && { echo "Usage: $0 filename"; exit 1; }
[ ! -f "$1" ] && { echo "File not found!"; exit 1; }

# Getting search word from user
read -p "Enter the word to search: " word1

# Counting occurrences
total=$(grep -ow "$word1" "$1" | wc -l)
[ $total -eq 0 ] && { echo "The word '$word1' was not found in the file."; exit 0; }

echo "Total occurrences of '$word1': $total"
echo "----------------------------------------"
grep -nw "$word1" "$1" | while read -r line; do
  lineno=$(echo "$line" | cut -d: -f1)
  content=$(echo "$line" | cut -d: -f2-)
  count=$(echo "$content" | grep -ow "$word1" | wc -l)
  echo "Line $lineno: $count occurrence(s)"
done
echo

# Get replacement word and replace
read -p "Enter the replacement word: " word2
newfile="modified_$1"
# sed stands for Stream Editor
sed -E "s/\b$word1\b/$word2/g" "$1" > "$newfile"
# sed "s/$word1/$word2/g" "$1" > "$newfile" (for MacOS)
echo "Replacement done. Modified content saved in '$newfile'."
```

# Check for partial and case-insensitive matches
partials=$(grep -o "$word1" "$1" | wc -l)
ignore_case=$(grep -iw "$word1" "$1" | wc -l)
[$partials -gt $total] && echo "Note: Partial matches of '$word1' exist in the file (not replaced)."
[$ignore_case -gt $total] && echo "Note: Matches exist if case sensitivity is ignored."

Example Output:



```
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh file2.c
Enter the word to search: factorial
Total occurrences of 'factorial':       5
------------------------------------------
Line 3:         1 occurrence(s)
Line 5:         2 occurrence(s)
Line 11:         1 occurrence(s)
Line 13:         1 occurrence(s)

Enter the replacement word: lcm
Replacement done. Modified content saved in 'modified_file2.c'.
```

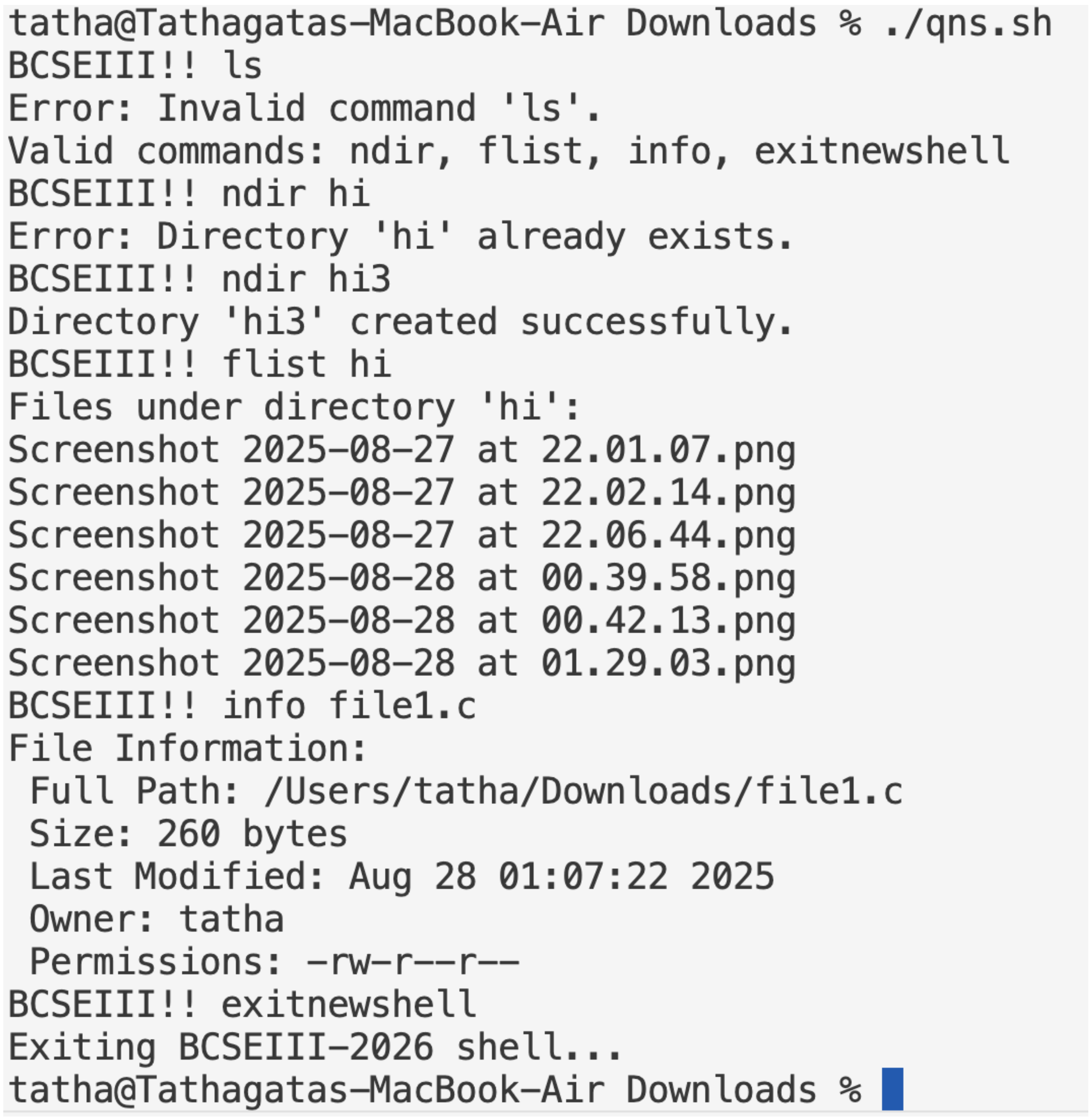
QUESTION 7:

Develop a Linux shell BCSEIII-2026, that will display a prompt, accept user commands and execute. Write appropriate routines (don't use linux commands) for the commands of BCSEIII-2026. The overview of the functions of BCSEIII-2026 is as follows: At startup, the shell will display the prompt string: BCSEIII!!, and ready to accept commands. A command line has the following syntax: command [argument/s]. When a valid command is entered by the user, the shell will execute the command. Commands may run in the foreground as well as in the background. The shell will return an appropriate error message when the command is invalid, or when there are problems with either the arguments or the execution of the command. The commands to be executed by BCSEIII-2026 are given below: (i) ndir directoryname <action: creates a new directory> (ii) flist directory name <action : all files (names) under the specified directory will be listed> (iii) info filename <action: displays essential information about the file specified, it must list the following: full path of the file, size of the file, last modification date, name of the creator, permissions> (iv) exitnewshell <action: this will quit the shell>

ANSWER :

```
#!/bin/bash
# Custom Shell: BCSEIII-2026
while true; do
  echo -n "BCSEIII!! "
  read cmd arg

  case $cmd in
     ndir)
        [ -z "$arg" ] && { echo "Error: Directory name missing."; continue; }
        [ -d "$arg" ] && { echo "Error: Directory '$arg' already exists."; continue;
}
        mkdir "$arg" 2>/dev/null && echo "Directory '$arg' created successfully."
|| echo "Error: Could not create directory '$arg'."
     ;;

     flist)
        [ -z "$arg" ] && { echo "Error: Directory name missing."; continue; }
        [ ! -d "$arg" ] && { echo "Error: '$arg' is not a valid directory."; continue;
}
        echo "Files under directory '$arg':"
        for file in "$arg"/*; do
           [ -f "$file" ] && echo "$(basename "$file")"
        done
     ;;

     info)
        [ -z "$arg" ] && { echo "Error: Filename missing."; continue; }
        [ ! -e "$arg" ] && { echo "Error: File '$arg' does not exist."; continue; }
        echo "File Information:"
        echo " Full Path: $(pwd)/$arg"
        echo " Size: $(stat -f %z "$arg") bytes"
        echo " Last Modified: $(stat -f %Sm "$arg")"
        echo " Owner: $(stat -f %Su "$arg")"
        echo " Permissions: $(stat -f %Sp "$arg")"
     ;;

     exitnewshell)
        echo "Exiting BCSEIII-2026 shell..."
        exit 0
    ;;

    *)
        echo "Error: Invalid command '$cmd'."
        echo "Valid commands: ndir, flist, info, exitnewshell"
    ;;
  esac
done
```

Example Output:



```
tatha@Tathagatas-MacBook-Air Downloads % ./qns.sh
BCSEIII!! ls
Error: Invalid command 'ls'.
Valid commands: ndir, flist, info, exitnewshell
BCSEIII!! ndir hi
Error: Directory 'hi' already exists.
BCSEIII!! ndir hi3
Directory 'hi3' created successfully.
BCSEIII!! flist hi
Files under directory 'hi':
Screenshot 2025-08-27 at 22.01.07.png
Screenshot 2025-08-27 at 22.02.14.png
Screenshot 2025-08-27 at 22.06.44.png
Screenshot 2025-08-28 at 00.39.58.png
Screenshot 2025-08-28 at 00.42.13.png
Screenshot 2025-08-28 at 01.29.03.png
BCSEIII!! info file1.c
File Information:
 Full Path: /Users/tatha/Downloads/file1.c
 Size: 260 bytes
 Last Modified: Aug 28 01:07:22 2025
 Owner: tatha
 Permissions: -rw-r--r--
BCSEIII!! exitnewshell
Exiting BCSEIII-2026 shell...
tatha@Tathagatas-MacBook-Air Downloads %
```