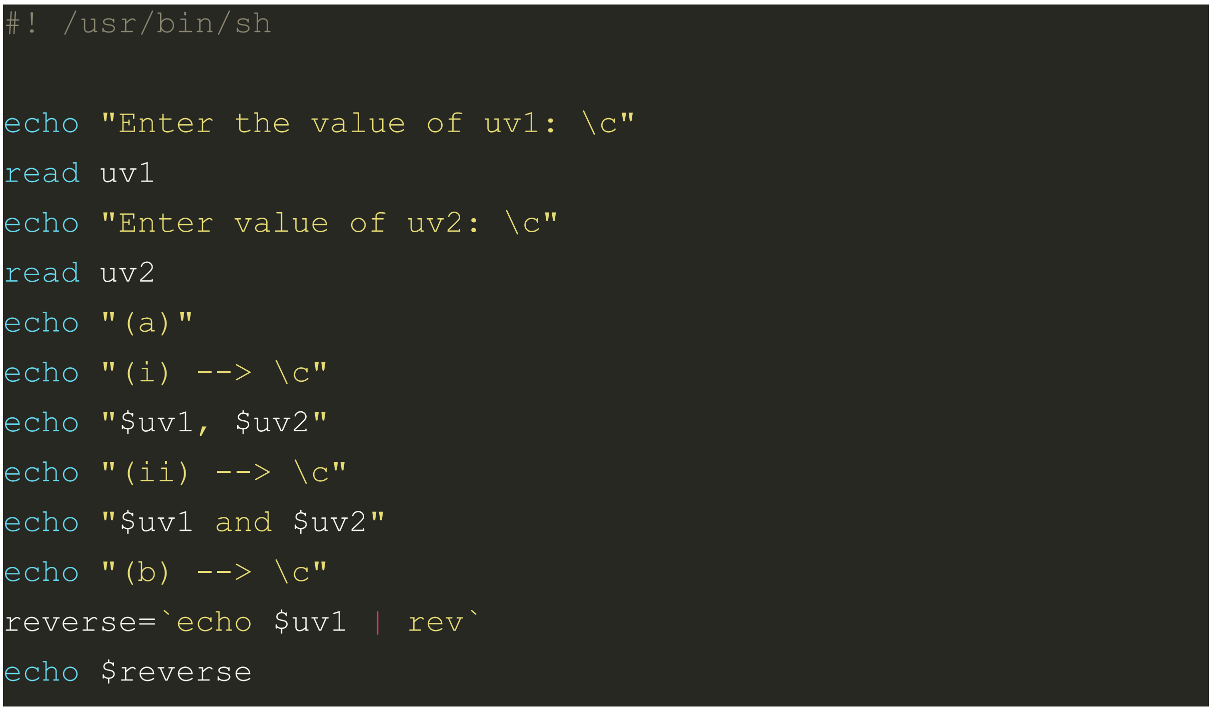
OS Lab Report
Assignment 1

Name : ROHIT SADHU
Roll No : 002010501074
Group : A3
Class : BCSE III
Semester : 5

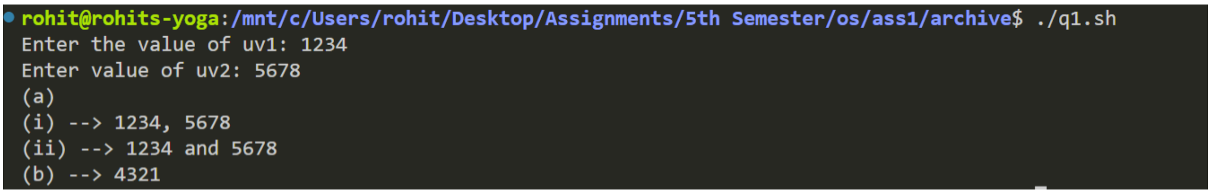
1. Write a shell script that has 2 user created variables, uv1 and uv2. Ask for the values of the
variables from the user and take in any values (real/integer/character) for the 2 variables. Test
the program for different types of uv1 and uv2.
(a) Print them as: (i) value of uv1 followed by value of uv2 separated by a comma and (ii)
value of uv2 followed by value of uv1 separated by the word “and”.
(b) Print the variables in reverse order [If uv1 is 1234, then output should be 4321]

Code :



```
#! /usr/bin/sh

echo "Enter the value of uv1: \c"
read uv1
echo "Enter value of uv2: \c"
read uv2
echo "(a)"
echo "(i) --> \c"
echo "$uv1, $uv2"
echo "(ii) --> \c"
echo "$uv1 and $uv2"
echo "(b) --> \c"
reverse=`echo $uv1 | rev`
echo $reverse
```

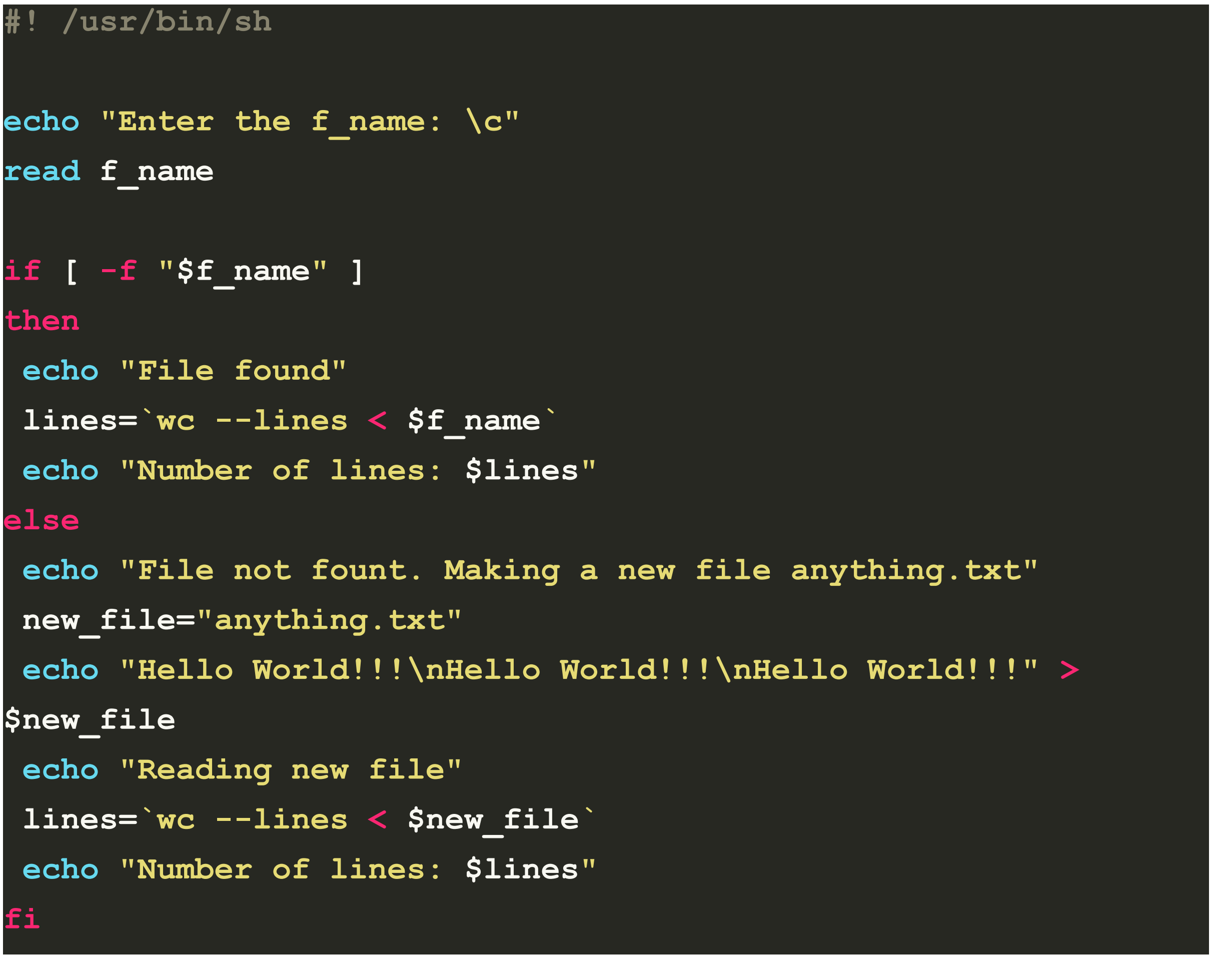
Output :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q1.sh
Enter the value of uv1: 1234
Enter value of uv2: 5678
(a)
(i) --> 1234, 5678
(ii) --> 1234 and 5678
(b) --> 4321
```

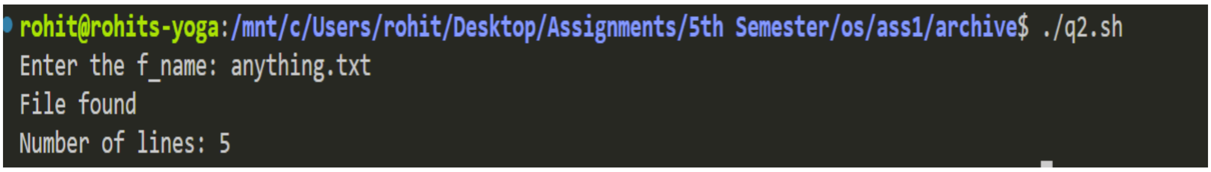
2. Write a shell script to count the number of lines in a file. Test if the file is present. If not, create and write.

Code :



```
#! /usr/bin/sh

echo "Enter the f_name: \c"
read f_name

if [ -f "$f_name" ]
then
 echo "File found"
 lines=`wc --lines < $f_name`
 echo "Number of lines: $lines"
else
 echo "File not fount. Making a new file anything.txt"
 new_file="anything.txt"
 echo "Hello World!!!\nHello World!!!\nHello World!!!" >
$new_file
 echo "Reading new file"
 lines=`wc --lines < $new_file`
 echo "Number of lines: $lines"
fi
```

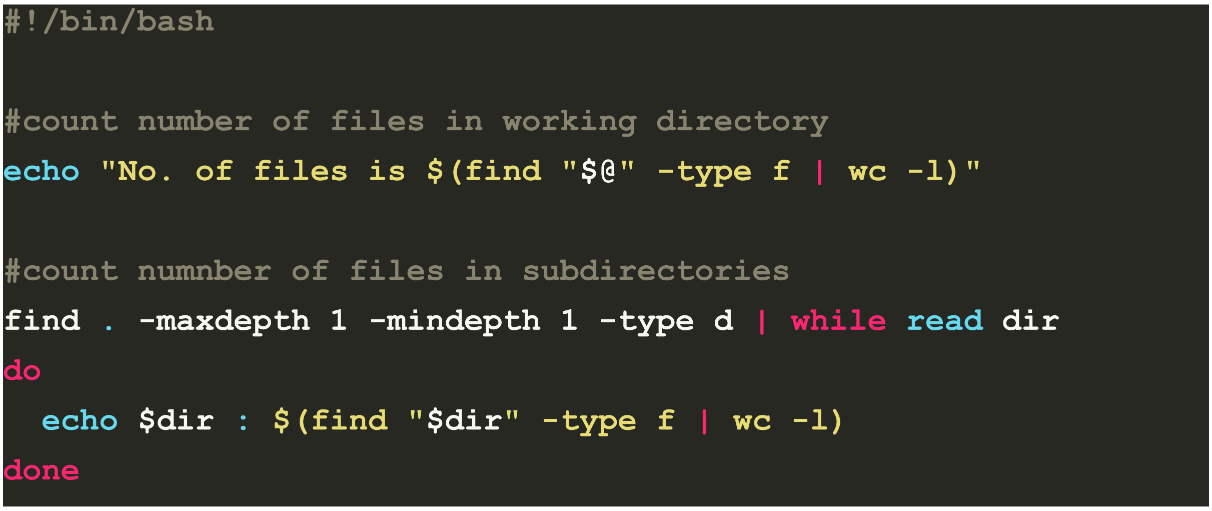
Output :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q2.sh
Enter the f_name: anything.txt
File found
Number of lines: 5
```

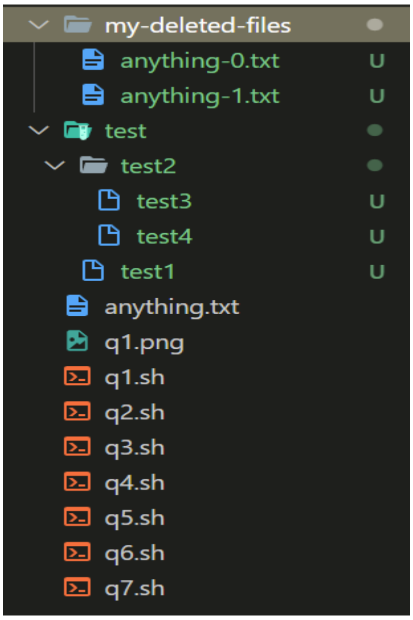
3. Write a shell script that counts the number of ordinary files (not directories) in the current working directory and its sub-directories. Repeat the count of files including the sub-directories that the current working directory has.

Code :



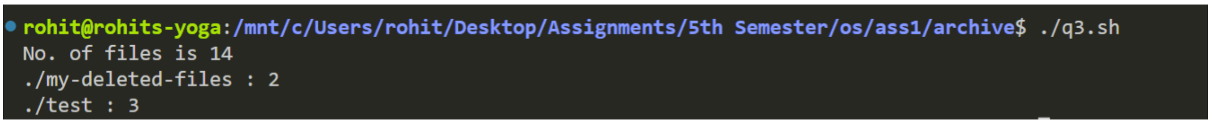
```
#!/bin/bash

#count number of files in working directory
echo "No. of files is $(find "$@" -type f | wc -l)"

#count numnber of files in subdirectories
find . -maxdepth 1 -mindepth 1 -type d | while read dir
do
  echo $dir : $(find "$dir" -type f | wc -l)
done
```

Test Directory structure :



```
v my-deleted-files
    anything-0.txt      U
    anything-1.txt      U
v test
  v test2
      test3             U
      test4             U
    test1               U
  anything.txt
  q1.png
  q1.sh
  q2.sh
  q3.sh
  q4.sh
  q5.sh
  q6.sh
  q7.sh
```

Output :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q3.sh
No. of files is 14
./my-deleted-files : 2
./test : 3
```

4. Write a shell program to duplicate the UNIX rm command with the following features:

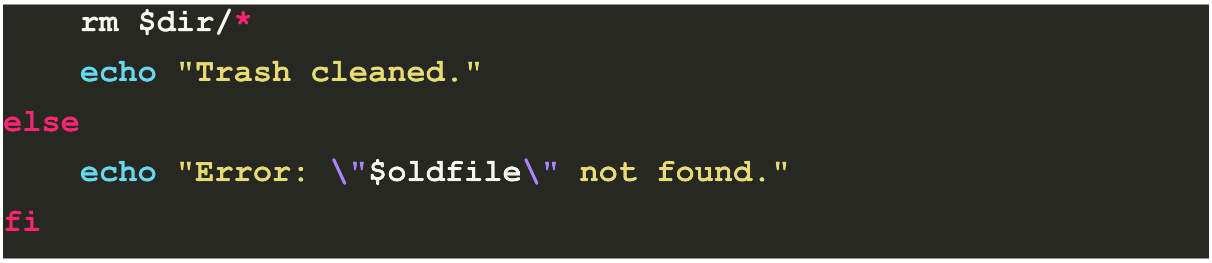
a. Instead of deleting the files, it will move them to a my-deleted-files directory. If the file already exists in the my-deleted-files directory, then the existing file (in the my-deleted-files) will have the version number zero (0) appended to it and the newly deleted file will have version number one (1) appended to it. Go on incrementing the version nos., if required.

b. The command will have a switch -c that will clear the entire my-deleted-files directory after asking for confirmation.

Code :

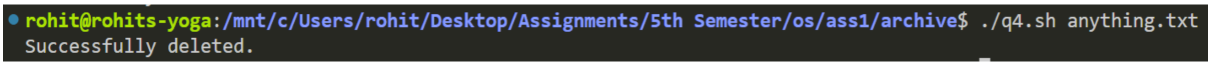
```
#! /usr/bin/bash

dir=my-deleted-files

# this is the switch or the file to be deleted
oldfile=$(basename -- $1)

# check if the file exists
if [ -f $oldfile ]
then
    # delete the shortest occurence of .* from back of file
    oldfilename="${oldfile%.*}"

    # delete the longest occurence of *. from front of file
    extension="${oldfile##*.}"

    # initialise the version as 0
    version=-1
    newfile="$oldfile"

    # change directory to trash folder
    cd $dir

    firstfile="${oldfilename}-0.${extension}"

    # check if some version of the file is already existing
    if [ -f $oldfile -o -f $firstfile ]; then
```

```
        # if already only one instance of the file exists in trash, assign version 0 to it
        if [ -f $oldfile ]; then
            mv $oldfile $firstfile
        fi

        # extract the file with the greatest version number
        for file in $(find . -type f -name "${oldfilename}*.${extension}")
        do
            filename="${file%.*}"
            curver=${filename#*-}
            if [ $curver -gt $version ]; then
                version=$curver
            fi
        done
    fi

    # return back to main directory
    cd ..

    # check if the version needs to be incremnted
    if [ $version -ge 0 ]; then
        version=$(($version + 1))
        # rename the old file
        newfile="${oldfilename}-${version}.${extension}"
        mv $oldfile $newfile
    fi

    # move the file inside the trash folder
    mv $newfile $dir
    echo "Successfully deleted."

elif [ "$oldfile" = "-c" ]
then
    # clear the trash
```

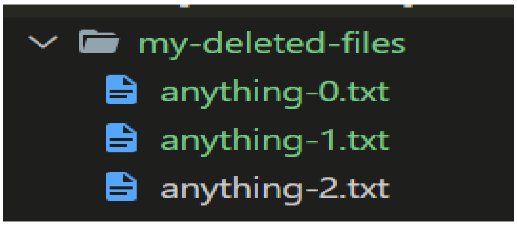


```
    rm $dir/*
    echo "Trash cleaned."
else
    echo "Error: \"$oldfile\" not found."
fi
```

Output :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q4.sh anything.txt
Successfully deleted.
```

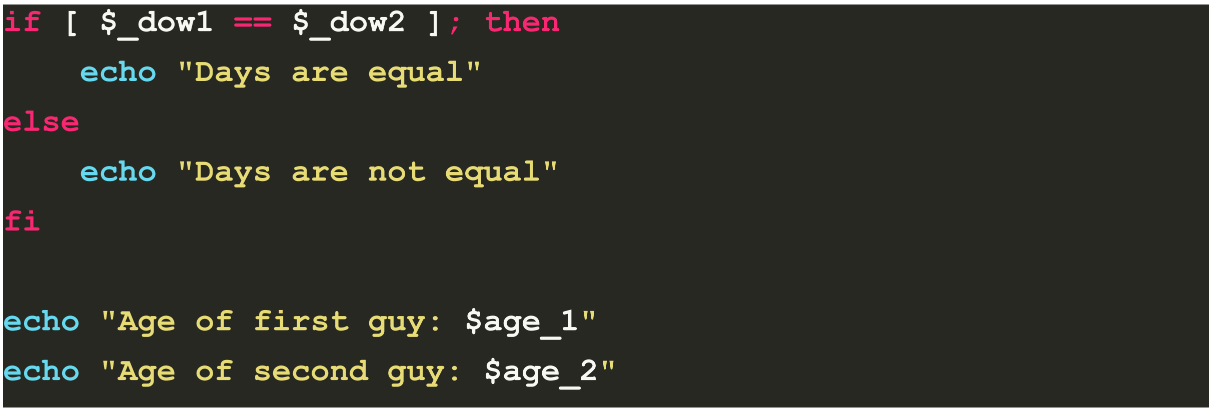


```
my-deleted-files
    anything-0.txt
    anything-1.txt
    anything-2.txt
```

5. Write a script called birthday_match.sh that takes two birthdays of the form DD/MM/YYYY
(e.g., 15/05/2000) and returns whether there is a match if the two people were born on the
same day of the week (e.g., Friday). And then find out the age/s in years/months/days.

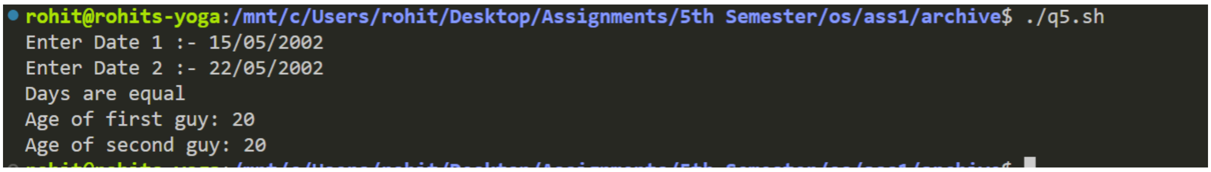
Code :

```
#! /usr/bin/bash

read -p "Enter Date 1 :- " date1
read -p "Enter Date 2 :- " date2
today_date_year=`date "+%Y"`
today_date_month=`date "+%m"`
today_date_day=`date "+%d"`

#evaluating the first day
day=${date1%%/*}
year=${date1##*/}
month=${date1#*/}
month=${month%/*}
_year_1=$(date --date="$month/$day/$year" '+%Y')

age_1=$(($today_date_year - $_year_1))

_dow1=$(date --date="$month/$day/$year" '+%A')

#evaluating the second day
day=${date2%%/*}
year=${date2##*/}
month=${date2#*/}
month=${month%/*}
_year_2=$(date --date="$month/$day/$year" '+%Y')

age_2=$(($today_date_year - $_year_2))

_dow2=$(date --date="$month/$day/$year" '+%A')
```



```bash
if [ $_dow1 == $_dow2 ]; then
    echo "Days are equal"
else
    echo "Days are not equal"
fi

echo "Age of first guy: $age_1"
echo "Age of second guy: $age_2"
```

Output :

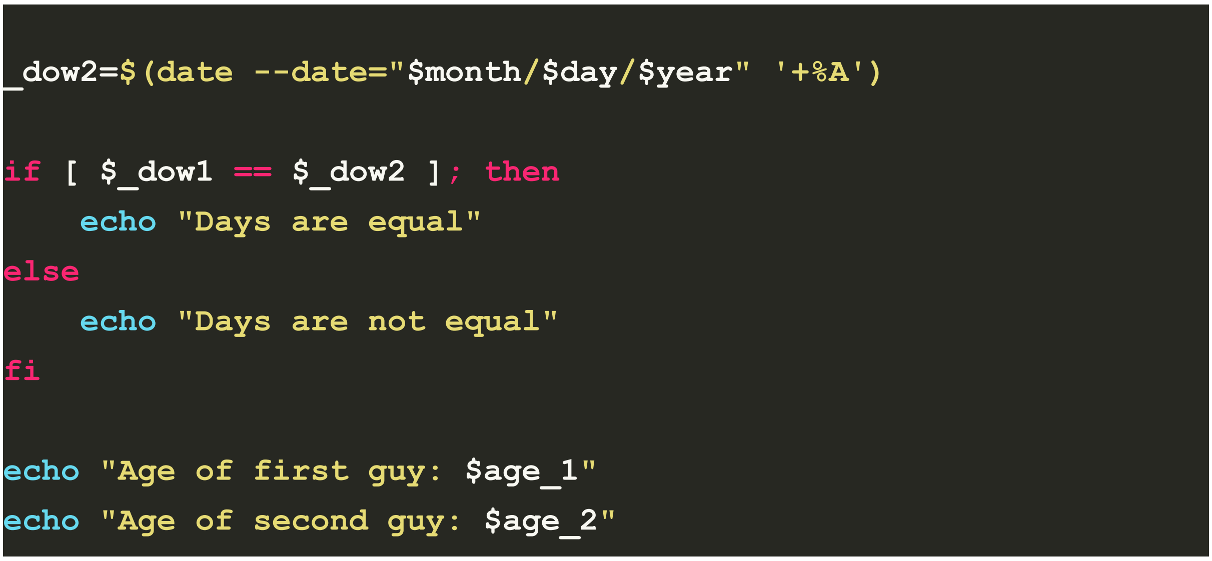


```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q5.sh
Enter Date 1 :- 15/05/2002
Enter Date 2 :- 22/05/2002
Days are equal
Age of first guy: 20
Age of second guy: 20
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$
```

6. Write a shell script that accepts a filename as an input and performs the following activities on the given file. The program 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. If the file contains the given word, the program will display (a) the number of occurrences of the word. The program is also required to display (b) the line number in which the word has occurred and no. of times the word has occurred in that line (Note: the word may occur more than once in a given line). If the file does not contain the word, an appropriate error message will be displayed.

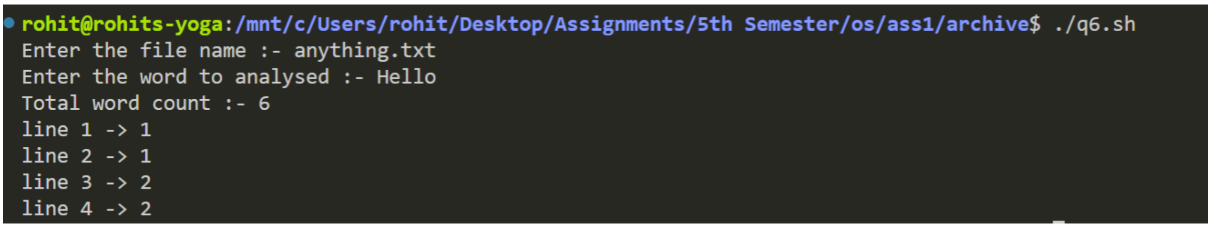
Code :

```
#! /usr/bin/bash

read -p "Enter Date 1 :- " date1
read -p "Enter Date 2 :- " date2
today_date_year=`date "+%Y"`
today_date_month=`date "+%m"`
today_date_day=`date "+%d"`

#evaluating the first day
day=${date1%%/*}
year=${date1##*/}
month=${date1#*/}
month=${month%/*}
_year_1=$(date --date="$month/$day/$year" '+%Y')

age_1=$(($today_date_year - $_year_1))

_dow1=$(date --date="$month/$day/$year" '+%A')

#evaluating the second day
day=${date2%%/*}
year=${date2##*/}
month=${date2#*/}
month=${month%/*}
_year_2=$(date --date="$month/$day/$year" '+%Y')

age_2=$(($today_date_year - $_year_2))
```



```
_dow2=$(date --date="$month/$day/$year" '+%A')

if [ $_dow1 == $_dow2 ]; then
    echo "Days are equal"
else
    echo "Days are not equal"
fi

echo "Age of first guy: $age_1"
echo "Age of second guy: $age_2"
```

Output :
a.txt :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q6.sh
Enter the file name :- anything.txt
Enter the word to analysed :- Hello
Total word count :- 6
line 1 -> 1
line 2 -> 1
line 3 -> 2
line 4 -> 2
```

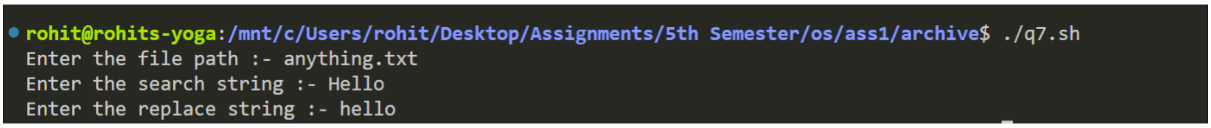
7. Extend the shell script written in (6) to perform the following task: User is asked to enter two different patterns or words. The first pattern will have to be matched with the contents of the file and replaced by the second pattern if match occurs. If the first pattern does not occur in the file, an appropriate error message will be displayed.

Code : Note that this code will be just appended after the end of q6.sh

```
#!/bin/bash
read -p "Enter the file path :- " file_path
read -p "Enter the search string :- " search
read -p "Enter the replace string :- " replace

word_count=$(grep -o -n $search $file_path | wc -l)

if [ "$word_count" -eq 0 ]
then
    echo "Word Not Found !"
    exit
fi

if [[ $search != "" && $replace != "" ]]
then
    sed -i "s/$search/$replace/g" $file_path
fi
```

Output :

a.txt before running script :



```
rohit@rohits-yoga:/mnt/c/Users/rohit/Desktop/Assignments/5th Semester/os/ass1/archive$ ./q7.sh
Enter the file path :- anything.txt
Enter the search string :- Hello
Enter the replace string :- hello
```