NAME - MAHFUJUL HAQUE	**CLASS ROLL - 002010501036**
CLASS - BCSE III	**SESSION - 2022-2023**
SEMESTER - FIRST	**ASSIGNMENT - 01**

QUESTION 01: 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]

APPROACH: To print the variables we use the **echo** function. Reversing has been done using **for** loop.

CODE:

```
read -p "Enter uv1:" uv1
read -p "Enter uv2:" uv2

echo "$uv1,$uv2"
echo "$uv2 and $uv1"

len1=${#uv1}
len2=${#uv2}

for((i=$len1-1;i>=0;--i))
do
    rev1="$rev1${uv1:$i:1}"
done

for((i=$len2-1;i>=0;--i))
do
    rev2="$rev2${uv2:$i:1}"
done

echo "uv1 in reverse order:$rev1"
echo "uv2 in reverse order:$rev2"
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502: bash q1.sh
Enter uv1:abc
Enter uv2:123
abc,123
123 and abc
uv1 in reverse order:cba
uv2 in reverse order:321
```

QUESTION 02: Write a shell script to count the number of lines in a file. Test if the file is present. If not, create and write.

APPROACH: To check whether the file exists or not, we use **-f option.** Number of lines has been obtained using **grep** command with **-c option.** We use **touch** to create the file in case it does not exist and append lines using a **for loop**.

CODE:

```
read -p "Enter filename:" file

if [ -f $file ]
then
    echo "File exists"
    lines=`grep -c "." $file`
    echo "Number of lines: $lines"
else
    echo "File does not exist. Creating new file..."
    read -p "Enter text(# to end input):" line
    touch $file
    while [[ $line != "#" ]]
    do
        echo $line >> $file
        read line
    done
fi
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q2.sh
Enter filename:file.txt
File exists
Number of lines: 3
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q2.sh
Enter filename:new.txt
File does not exist. Creating new file...
Enter text(# to end input):This is another file.
We add another line.
Lets end.
#
```

QUESTION 03: 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 subdirectories that the current working directory has.

APPROACH: This can be done by **recursively** calling a function for all the subdirectories. Two functions were generated, one counted **only the files**, and another counted the **number of files as well as directories**.

CODE:

```
count_files_only()
{
    if (( $# == 0 ))
    then
        echo 0
    else
        path="$1"
        no_of_files=0
        for file in "$path"/*
        do
            if [ -d "$file" ]
            then
                rec_return=$(count_files_only "$file/")
                no_of_files=$(($no_of_files + $rec_return))
            elif [ -f "$file" ]
            then
                no_of_files=$(($no_of_files + 1))
            fi
        done

        echo $no_of_files
    fi
}

count_files_with_subdir()
{
    if (( $# == 0 ))
    then
        echo 0
    else
        path="$1"
        no_of_files=0
        for file in "$path"/*
        do
            if [ -d "$file" ]
            then
                no_of_files=$(($no_of_files + 1))
                rec_return=$(count_files_with_subdir "$file/")
                no_of_files=$(($no_of_files + $rec_return))
            elif [ -f "$file" ]
            then
                no_of_files=$(($no_of_files + 1))
            fi
        done

        echo $no_of_files
```

```
    fi
}

ans1=$(count_files_only ".")
echo "Number of files only: $ans1"
ans2=$(count_files_with_subdir ".")
echo "Number of files with sub-directories: $ans2"
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q3.sh
Number of files only: 16
Number of files with sub-directories: 20
```

QUESTION 04: 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.

APPROACH: First we check whether the **my_deleted_files** directory exists or not. If it does not, we create one. Then we check whether the file to be deleted exists or not. If it exists, we find a match for it in the **my_deleted_files** directory. If there exists one, we append version number zero to it and move the original file with a version number 1. If the match already has a version number zero with it, we count total number of versions and then move the original file with count as version number appended. When called with option **-c**, we delete the directory.

CODE:

```
argc=$#
REC_BIN="./my_deleted_files"
if ! [ -d $REC_BIN ]
then
    mkdir -p $REC_BIN
fi

if [ $argc == 0 ]
then
    echo "Error: No input"
elif [ $argc -eq 2 ]
then
    echo "Error: Multiple inputs"
else
    if [ $1 == "-c" ]
    then
        read -p "Do you want to clear my_deleted_files? (y/n): " conf
        if [ $conf == "y" ]
        then
            rm -rf $REC_BIN
```

```
            echo "my_deleted_files Cleared!!!"
        fi
    else
        file=$1
        if ! [ -f $file ]
        then
            echo "Error: File does not exist..."
        else
            if [ -f $REC_BIN"/"${file%.*}"_0."${file##*.} ]
            then
                count=0
                while [ -f $REC_BIN"/"${file%.*}"_"$count"."$
{file##*.} ]
                do
                    count=$(($count + 1))
                done
                newfile=${file%.*}"_"$count"."${file##*.}
                mv $file $REC_BIN"/"$newfile
            elif [ -f $REC_BIN"/"$file ]
            then
                mv $REC_BIN"/"$file $REC_BIN"/"${file%.*}"_0."$
{file##*.}
                mv $file $REC_BIN"/"${file%.*}"_1."${file##*.}
            else
                mv $file $REC_BIN"/"$file
            fi
            echo "File deleted succesfully..."
        fi
    fi

fi
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q4.sh my.txt
File deleted succesfully...
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q4.sh -c
Do you want to clear my_deleted_files? (y/n): y
my_deleted_files Cleared!!!
```

QUESTION 05: 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.

APPROACH: **date** command is used to obtain the week days. To calculate age, we find the age in seconds and convert it into years, months, days format by division.

CODE:

```
conv() {
    T=$1
    Y=$((T/60/60/24/365))
    Mo=$((T/60/60/24%365/30))
    D=$((T/60/60/24%365%30))

    (( $Y > 0 )) && res="$res$Y years "
    (( $M > 0 )) && res="$res$M months "
    (( $D > 0 )) && res="$res$D days"
    echo $res
}

read -p "Enter first birthday: " b1
read -p "Enter second birthday: " b2

IFS='/' read -r -a dateArr1 <<< $b1
IFS='/' read -r -a dateArr2 <<< $b2

# temp=${dateArr1[0]}
# dateArr1[0]=${dateArr1[1]}
# dateArr1[1]=$temp

# temp=${dateArr2[0]}
# dateArr2[0]=${dateArr2[1]}
# dateArr2[1]=$temp

b1="${dateArr1[1]}/${dateArr1[0]}/${dateArr1[2]}"
b2="${dateArr2[1]}/${dateArr2[0]}/${dateArr2[2]}"

day1=$(date +%a --date=$b1)
day2=$(date +%a  --date=$b2)

echo $day1
echo $day2

if [ $day1 == $day2 ]
then
    echo "Day matches"
else
    echo "Day does not match"
fi

cur=$(date +%s)
date1=$(date +%s --date=$b1)
date2=$(date +%s  --date=$b2)

a1=$((cur-date1))
a2=$((cur-date2))
```

```
age1=$(conv $a1)
age2=$(conv $a2)

echo "Age of first person: $age1"
echo "Age of second person: $age2"
```

OUPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/Desktop/OS_Lab$ bash q5.sh
Enter first birthday: 10/12/2002
Enter second birthday: 04/12/2001
Tue
Tue
Day matches
Age of first person: 19 years 8 months 27 days
Age of second person: 20 years 9 months 3 days
```

QUESTION 06: Write a shell script that accepts a file name 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.

APPROACH: **grep** command has been used to obtain the number of lines in the file with the given string. To find the number of occurences in each line, we iterate through the lines of the file using **while read** loop.

CODE:

```
read -p "Enter file name: " file
read -p "Enter string to search for: " str

total_cnt=`grep -w -c $str $file`
echo "No. of lines containing given string: $total_cnt"
i=1
if [ $total_cnt -ne 0 ]
then
    while read -r line
    do
        cnt=`grep -o $str <<< $line | wc -l`
        if [ $cnt -ne 0 ]
        then
            echo "Occurences in line $i: $cnt"
        fi
        i=$((i+1))
    done < $file
else
    echo "No lines contain given string."
fi
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/   File:
Desktop/OS_Lab$ bash q6.sh           sky is blue
Enter file name: new.txt             sea is blue
Enter string to search for: blue     blue is blue
No. of lines containing given        red is red
string: 3
Occurences in line 1: 1
Occurences in line 2: 1
Occurences in line 3: 2
```

QUESTION 07: 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 a match occurs. If the first pattern does not occur in the file, an appropriate error message will be displayed.

APPROACH: To replace the string with another one, we use the **sed** command.

CODE:

```
read -p "Enter file name: " file
read -p "Enter pattern 1: " str1
read -p "Enter pattern 2: " str2

total_cnt=`grep -w -c $str1 $file`
if [ $total_cnt -ne 0 ]
then
    text=`sed -i "s/$str1/$str2/g" $file`
else
    echo "Error: No lines contain given string."
fi
```

OUTPUT:

```
mahfujul@mahfujul-Inspiron-5502:~/   File:
Desktop/OS_Lab$ bash q7.sh           sky is green
Enter file name: new.txt             sea is green
Enter pattern 1: blue                green is green
Enter pattern 2: green               red is red
```