|  |  |
| --- | --- |
| **NAME:** Kanko Ghosh  **CLASS:** BCSE III  **SEM:** FIRST | **CLASS ROLL NO.:** 002010501035  **SESSION:** 2022-2023  **ASSIGNMENT SET:**  1 |

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]

**Solution**: 2 numbers are taken as input. Using for loop, the string is reversed; though “**rev**” function could be used to reverse the string.

**Code**:

echo "Program 1"

echo -n "Enter a variable: "

read uv1

echo -n "Enter another variable: "

read uv2

echo -n "Data Separated by comma: "

echo $uv1,$uv2

echo -n "Data Separated by comma: "

echo $uv2 and $uv1

len1=${#uv1}

len2=${#uv2}

for (( i=$len1-1; i>=0; i-- ));

do

    revstr=$revstr${uv1:$i:1}

done

for (( i=$len2-1; i>=0; i-- ));

do

    revstr2=$revstr2${uv2:$i:1}

done

echo Reverse of Input 1: $revstr

echo Reverse of Input 2: $revstr2

**Output**:

Program 1

Enter a variable: 123

Enter another variable: 234

Data Separated by comma: 123,234

Data Separated by comma: 234 and 123

Reverse of Input 1: 321

Reverse of Input 2: 432

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.

**Solution**: Presence of file in the directory is checked using **–f options** inside the if condition. If the file is present, the number of lines can be found using **“wc -l”** command. If the file is not present, the file must be created, followed by taking input data in a loop until a terminal character, i.e. **“$”** comes.

**Code:**

echo -n "Enter file name: "

read FNAME

if [ -f "$FNAME" ]

then

    echo -n "Number of lines in the file is: "

    echo `cat $FNAME | wc -l`

else

    echo "File $FNAME does not exist"

    echo "Creating a file with filename $FNAME"

    echo Enter Data, press $ to terminate

    touch $FNAME

    read data

    while [[ $data != "$" ]]

    do

        echo $data >> $FNAME

        read data

    done

fi

**Output**:

kanko@DESKTOP-14TFPUO:/mnt/f/5thsem/OS Lab Assg$ ./q1.sh

Program 1

Enter a variable: 123

Enter another variable: 234

Data Separated by comma: 123,234

hello

brilliant

1234

tata

$

Kanko@DESKTOP-14TFPUO:/mnt/f/5thsem/OS Lab Assg$ ./q2.sh

Enter file name: abc.txt

Number of lines in the file is: 4

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 subdirectories that the current working directory has.

**Solution**: 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.

count\_files()

{

    if [ $# -eq 0 ]

    then

        echo 0

    else

        PATH="$1"

        no\_of\_files=0

        for file in "$PATH"/\*

        do

            if [ -d "$file" ]

            then

                rec\_return=$(count\_files "$file/")

                no\_of\_files=$(($no\_of\_files + $rec\_return))

            else

                no\_of\_files=$(($no\_of\_files + 1))

            fi

        done

        echo $no\_of\_files

    fi

}

count\_files\_with\_dir()

{

    if [ $# -eq 0 ]

    then

        echo 0

    else

        PATH="$1"

        no\_of\_files=0

        for file in "$PATH"/\*

        do

            no\_of\_files=$(($no\_of\_files + 1))

            if [ -d "$file" ]

            then

                rec\_return=$(count\_files\_with\_dir "$file/")

                no\_of\_files=$(($no\_of\_files + $rec\_return))

            fi

        done

        echo $no\_of\_files

    fi

}

ans=$(count\_files ".")

echo "Number of files: $ans"

ans=$(count\_files\_with\_dir ".")

echo "Number of files and directories: $ans"

Output:

swaha@DESKTOP-14TFPUO:/mnt/f/5thsem/OS Lab Assg$ ./q3.sh

Number of files: 31

Number of files and directories: 34

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.

**Solution**: The count of the next file to be printed, is done by by for loop. on using –c, the whole “my-deleted-files” folder is deleted.

**Code**:

argc=$#

if [ $argc -eq 0 ]

then

    echo "NO INPUT"

elif [ $argc -eq 2 ]

then

    echo "MORE THAN 1 INPUT"

else

    REC\_BIN="./my\_deleted\_files"

    if [ $1 == "-c" ]

    then

        if ! [ -d $REC\_BIN ]

        then

            echo "Recycle bin doesnt exist, so it is already empty!"

        else

            echo "Recycle bin exists"

            #clear it

            echo "Are u sure u want to clear? (Y/N)"

            read conf

            if [ $conf == "Y" ]

            then

                rm -rf $REC\_BIN

            fi

        fi

    else

        FNAME=$1

        if ! [ -f $FNAME ]

        then

            echo "File Doesnt Exist, invalid input"

        else

            if ! [ -d $REC\_BIN ]

            then

                echo "Recycle bin not exist, created!"

                mkdir -p $REC\_BIN

                mv $FNAME $REC\_BIN"/"$FNAME

                echo "File deleted"

            else

                echo "Recycle bin exists"

                if [ -f $REC\_BIN"/"${FNAME%.\*}"(0)."${FNAME##\*.} ]

                then

                    count=0

                    while [ -f $REC\_BIN"/"${FNAME%.\*}"("$count")."${FNAME##\*.} ]

                    do

                        count=$(($count + 1))

                    done

                    newFNAME=${FNAME%.\*}"("$count")."${FNAME##\*.}

                    mv $FNAME $REC\_BIN"/"$newFNAME

                elif [ -f $REC\_BIN"/"$FNAME ]

                then

                    mv $REC\_BIN"/"$FNAME $REC\_BIN"/"${FNAME%.\*}"(0)."${FNAME##\*.}

                    mv $FNAME $REC\_BIN"/"${FNAME%.\*}"(1)."${FNAME##\*.}

                else

                    mv $FNAME $REC\_BIN"/"$FNAME

                fi

                echo "File deleted"

            fi

        fi

    fi

fi

**Output**

$ touch txt.txt

$ ./q4.sh txt.txt

Recycle bin exists

File deleted

$ ./q4.sh txt.txt

File Doesnt Exist, invalid input

$ touch txt.txt

$ ./q4.sh txt.txt

Recycle bin exists

File deleted

$ ./q4.sh -c

Recycle bin exists

Are u sure u want to clear? (Y/N)

Y

$ ./q4.sh txt.txt

File Doesnt Exist, invalid input

$ touch txt.txt

$ ./q4.sh txt.txt

Recycle bin not exist, created!

File deleted

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.

**Solution**: ‘date’ commands can be used to retrieve the days, and can also detect the dates that are invalid. A function was created to generate the age from a given date.

**Code**:

date\_disp() {

    T=$1

    Y=$((T/60/60/24/365))

    Mo=$((T/60/60/24%365/30))

    D=$((T/60/60/24%365%30))

    H=$((T/60/60%24))

    M=$((T/60%60))

    S=$((T%60))

    (( $Y > 0 )) && res="$res$Y years, "

    (( $Mo > 0 )) && res="$res$Mo months, "

    (( $D > 0 )) && res="$res$D days, "

    (( $H > 0 )) && res="$res$H hours, "

    (( $M > 0 )) && res="$res$M minutes, "

    (( $S > 0 )) && res="$res$S seconds"

    echo $res

}

echo -n "enter date number 1(DD/MM/YYYY): "

read dt1

ddmm1=${dt1%/\*}

dd1=${ddmm1%/\*}

mm1=${ddmm1##\*/}

yyyy1=${dt1##\*/}

date1=`date -d $mm1/$dd1/$yyyy1`

echo -n "enter date number 2(DD/MM/YYYY): "

read dt2

ddmm2=${dt2%/\*}

dd2=${ddmm2%/\*}

mm2=${ddmm2##\*/}

yyyy2=${dt2##\*/}

date2=`date -d $mm2/$dd2/$yyyy2`

choice1=${date1:0:2}

case $choice1 in

    "Mo") echo Person one was born in: Monday ;;

    "Tu") echo Person one was born in: Tuesday ;;

    "We") echo Person one was born in: Wednesday ;;

    "Th") echo Person one was born in: Thursday ;;

    "Fr") echo Person one was born in: Friday ;;

    "Sa") echo Person one was born in: Saturday ;;

    "Su") echo Person one was born in: Sunday ;;

    "da") echo Date is invalid ;;

esac

choice2=${date2:0:2}

case $choice2 in

    "Mo") echo Person two was born in: Monday ;;

    "Tu") echo Person two was born in: Tuesday ;;

    "We") echo Person two was born in: Wednesday ;;

    "Th") echo Person two was born in: Thursday ;;

    "Fr") echo Person two was born in: Friday ;;

    "Sa") echo Person two was born in: Saturday ;;

    "Su") echo Person two was born in: Sunday ;;

    "da") echo Date is invalid ;;

esac

if [ $choice1 != "da" ] && [ $choice2 != "da" ]

then

    if [ $choice1 == $choice2 ]

    then

        echo "Same day"

    else

        echo "different day"

    fi

fi

if [ $choice1 != "da" ]

then

    echo -n "Age of person 1 is: "

    temp=`date +%s.%N -d $mm1/$dd1/$yyyy1`

    cur=`date +%s.%N`

    temp=$(echo "$cur - $temp" | bc)

    echo `date\_disp ${temp%.\*}`

fi

if [ $choice2 != "da" ]

then

    echo -n "Age of person 2 is: "

    temp=`date +%s.%N -d $mm2/$dd2/$yyyy2`

    cur=`date +%s.%N`

    temp=$(echo "$cur - $temp" | bc)

    echo `date\_disp ${temp%.\*}`

fi

**Output**:

enter date number 1(DD/MM/YYYY): 11/11/2001

enter date number 2(DD/MM/YYYY): 20/08/2022

Person one was born in: Sunday

Person two was born in: Saturday

different day

Age of person 1 is: 20 years, 9 months, 26 days, 20 hours, 22 minutes, 12 seconds

Age of person 2 is: 9 days, 20 hours, 22 minutes, 12 seconds

6. 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.

**Solution**: grep command was used to find the number of words in the file as well as in each line. awk was used to extract out the lines from the file, and count is shown for each line. File name is taken as command line argument

**Code**:

argc=$#

if [ $argc -eq 0 ]

then

    echo "NO INPUT"

elif [ $argc -eq 2 ]

then

    echo "MORE THAN 1 INPUT"

else

    FNAME=$1

    if ! [ -f $FNAME ]

    then

        echo "File Doesnt Exist, invalid input"

    else

        echo -n "Enter the word to find: "

        read str

        linecnt=`wc -l < $FNAME`

        echo "Number of lines in file: $linecnt"

        echo "Number of lines having string \"$str\": `grep -w $str $FNAME | wc -l`"

        for (( i=1; i<=$linecnt; i++ ))

        do

            cnt=`awk "NR==$i {print}" $FNAME | grep -o -w $str | wc -l`

            if [ $cnt -ne 0 ]

            then

                echo "count in line $i: $cnt"

            fi

        done

    fi

fi

|  |  |
| --- | --- |
| Output  $ ./q6.sh txt.txt  Enter the word to find: helllo  Number of lines in file: 4  Number of lines having string "helllo": 3  count in line 1: 1  count in line 3: 2  count in line 4: 1 | File:  helllo  asdasd  helllo and helllo  helllo |

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

**Solution**: Same as previous, the **–w** option was removed in grep command, since we are not finding whole matching words. Replace is done by “**sed**” command

**Output**:

$ ./q7.sh txt.txt

Enter the pattern to find: lll

Enter the word to replace with: ll

Number of lines in file: 4

Number of lines having string "lll": 3

count in line 1: 1

count in line 3: 2

count in line 4: 1

Total Count: 4

File Saved!

**Additional Assignment**

**Solution Approach:** the username is already defined in USER variable. The greetings is done based on the time, which can be known based on the time from “date” command. All the files in the current directory is listed by “ls” command. The disc usage is found using “df” command. The whole program is a menu driven program

**Code:**

func\_1() {

    echo "hello, $USER!!" >&2

    hr=`date +%H`

    if [ $hr -lt 6 ]

    then

        echo Good Evening

    elif [ $hr -lt 12 ]

    then

        echo Good Morning

    elif [ $hr -lt 18 ]

    then

        echo Good Afternoon

    else

        echo Good Evening

    fi

}

func\_2() {

    size=$1

    cnt=`ls -l | wc -l`

    echo -e "FileName\tSize" >&2

    echo "--------------------------------------------------" >&2

    for (( i=2; i<=$(($cnt)); i++ ))

    do

        if [ `ls -l | awk "NR==$i {print $5}" | awk '{print $5}'` -gt $size ]

        then

            echo -n `ls -l | awk "NR==$i {print $5}" | awk '{print $9}'` >&2

            echo -e -n '\t' >&2

            echo `ls -l | awk "NR==$i {print $5}" | awk '{print $5}'` >&2

        fi

    done

}

func\_3() {

    size=`df -h | wc -l`

    echo -e "Partition\tTotal\tAvlbl\tUsed\t%used" >&2

    echo "--------------------------------------------------" >&2

    for (( i=$(($size-3)); i<=$size; i++ ))

    do

        var=`df -h | awk "NR==$i {print}" | awk '{print $6}'`

        echo -n -e $var"\t\t" >&2

        var=`df -h | awk "NR==$i {print}" | awk '{print $2}'`

        echo -n -e $var"\t" >&2

        var=`df -h | awk "NR==$i {print}" | awk '{print $4}'`

        echo -n -e $var"\t" >&2

        var=`df -h | awk "NR==$i {print}" | awk '{print $3}'`

        echo -n -e $var"\t" >&2

        var=`df -h | awk "NR==$i {print}" | awk '{print $5}'`

        echo -e $var"\t" >&2

    done

}

func\_4() {

    linecnt=`cat log.txt | wc -l`

    for (( i=1; i<=$linecnt; i++ ))

    do

        echo `awk "NR==$i {print}" "log.txt"` >&2

    done

}

    echo "      +--------+ "

    echo "      |  MENU  | "

    echo "      +--------+ "

    echo "Enter the number: "

    echo "    1. Display greetings"

    echo "    2. List large files"

    echo "    3. Disk usage"

    echo "    4. View log file"

    echo "    5. exit"

while true

do

    echo -n ">>>  "

    read choice

    case $choice in

    "1")

        echo $(func\_1)

        ;;

    "2")

        echo -n "Enter number of bytes: " >&2

        read size

        echo $(func\_2 $size)

        ;;

    "3")

        echo $(func\_3)

        ;;

    "4")

        echo $(func\_4)

        ;;

    "5")

        echo Exiting..

        exit

        ;;

    \*)

        printf "Invalid Statement\n"

    esac

    case $choice in

    "1") menu="GREET USER" ;;

    "2") menu="LARGE LIST" ;;

    "3") menu="DISK USAGE" ;;

    "4") menu="VIEW LOG" ;;

    "5") menu="EXIT PROG" ;;

    \*) menu="INVALID INP" ;;

    esac

    curDate=`date`

    echo "$USER % $menu % $curDate" >> log.txt

done

Output

swaha@DESKTOP-14TFPUO:/mnt/f/5thsem/OS Lab Assg$ ./1035\_kanko.sh

+--------+

| MENU |

+--------+

Enter the number:

1. Display greetings

2. List large files

3. Disk usage

4. View log file

5. exit

>>> 1

hello, swaha!!

Good Evening

>>> 2

Enter number of bytes: 10000

FileName Size

--------------------------------------------------

Assg1.docx 40681

Assg1.pdf 91390

BASH 1759034

temp1.exe 51849

temp2.exe 84240

>>> 3

Partition Total Avlbl Used %used

--------------------------------------------------

/mnt/c 119G 35G 85G 72%

/mnt/d 297G 237G 60G 21%

/mnt/e 323G 188G 135G 42%

/mnt/f 313G 262G 52G 17%

>>> 4

swaha % GREET USER % Mon Aug 29 22:36:47 IST 2022

swaha % LARGE LIST % Mon Aug 29 22:36:57 IST 2022

swaha % DISK USAGE % Mon Aug 29 22:36:59 IST 2022

>>> 4

swaha % GREET USER % Mon Aug 29 22:36:47 IST 2022

swaha % LARGE LIST % Mon Aug 29 22:36:57 IST 2022

swaha % DISK USAGE % Mon Aug 29 22:36:59 IST 2022

swaha % VIEW LOG % Mon Aug 29 22:37:01 IST 2022

>>> 5

Exiting..