```
************************GTU2************************
#!/usr/bin/env bash
# GTU2b.sh This script will check whether entered number of string is pelindrome or not.
#(AIO script for numbers, strings, including special characters... So GTU21 and GTU22 is included here.)
# Code written By: Rushyang Darji
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts" for regular updates
# and more scripts.

echo -n "Enter string: "
read string

N=${#string}    # Total no of characters of a string.

mid=$(($N/2))
i=0

while [ $i -lt $mid ]
do
        if [ "${string:$i:1}" != "${string: -$(($i+1)):1}" ];   then
# ${string:$i:1} will check 1 character from "ith" position from front. Note that i starts from 0.
# ${string: -$(($i+1)):1} will move from back. As i progresses, -$(($i+1)) value gets near to i.
# As soon as Any two characters are found unmatched, It will prompt as not pelindrome, and exit quickly.
                echo "String is not a pelindrome"
                exit
        else
                i=$(($i+1))
        fi
done

echo "String is pelindrome"

: << -- OUTPUT
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2b.sh
Enter string: rushyang
String is not a pelindrome
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2b.sh
Enter string: 1234321
String is pelindrome
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2b.sh
Enter string: 12abba21
String is pelindrome
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2b.sh
Enter string: qwejkllkjewq
String is pelindrome
--

#!/usr/bin/env bash
# GTU2c.sh Accept number and check the number is even or odd, finds the length of the number,
# sum of the digits in the number.
# Code written By: Rushyang Darji
# Last Build: 10.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

echo -n "Enter Number: "
read no

# Even or odd
i=0
no1=$((${no: -$(($i+1)):1})) # Grabs last digit of no and stores into no1.

a=$(expr "$no1" % 2)  # or `expr "$no1" % 2`
if test "$a" -eq "0"; then
        echo "Number is Even"
else
        echo "Number is Odd"
fi
```

```
# Length of number
count=${#no}
sum=0
echo "No of Digits is: $count"

# Sum of all digits
while [ $count -gt 0 ]
do
        sum=`expr $sum + $((${no:$i:1}))`
        count=$(($count-1))
        i=$(($i+1))
done


echo "Sum of all digits: $sum"

: << --
                OUTPUT
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2c.sh
Enter Number: 123546
Number is Even
No of Digits is: 6
Sum of all digits: 21
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2c.sh
Enter Number: 8634597
Number is Odd
No of Digits is: 7
Sum of all digits: 42
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2c.sh
Enter Number: 0
Number is Even
No of Digits is: 1
Sum of all digits: 0
--

#!/usr/bin/env bash
# GTU2d: Accept strings and replace a string by another string.
# Code written By: Rushyang Darji
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

echo "Enter the Main string: "
read string

temp=$(mktemp)          # mktemp makes temporary file.
echo $string > $temp
echo "You entered..."   # just making sure, whether making temp was successful or not.
cat $temp

echo "Enter sub-string you want to replace..."
read str1

echo "Enter sub-string you want to relace with..."
read str2

sed -n s/$str1/$str2/gp < $temp
# s stands for "substitution",
# $str1 is what should be replaced. $str2 is from what $str1 should be replaced.
# g stands for "Global". Without it, only first match of $str1 will be replaced with $str2,
# and rest of $str1 will stay as it was.

rm $temp # Removing temp file.

: << --
        OUTPUT
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU2d.sh
```

```
Enter the Main string:
The old fox jumps over the big rock.
You entered...
The old fox jumps over the big rock.
Enter sub-string you want to replace...
fox
Enter sub-string you want to relace with...
frog
The old frog jumps over the big rock.
--

#!/usr/bin/env bash
# GTU2e.sh Accept filename and displays last modification time if file exists,
# otherwise display appropriate message.
# Code written By: Rushyang Darji
# Last Build: 10.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.


echo "Enter the filename"
read FILE

if [ -f $FILE ]; then
        echo "The file exists."
        echo "Last modification time is."
        ls -l $FILE | awk '{print $6" "$7}'
# prints 6th and 7th column from tabular result of ls -l
else
        echo "The file does not exist"
fi


#!/usr/bin/env bash
# GTU 2: Fetch the data from a file and display data into another file in reverse order
# Code written By: Rushyang Darji
# Last Build: 10.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
ls
echo "Enter a filename: "
read FILE

temp=$(mktemp tmp.XXXXX)
if [ -f $FILE ]; then
        rev $FILE > $temp
        echo "Data successfully fetched into $temp"
        cat $temp
else
        echo "File does not exist"
fi

rm -i $temp
***********************GTU3***********************
#!/usr/bin/env bash
# GTU3: Write a script to find the global complete path for any file.
# Code written By: Rushyang Darji
# Last Updated: 19.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

echo -e "Enter the filename to search in current directory: \c"
read FILE

args=`find . -name $FILE | xargs`
# xargs builds arguments from find, for using in "for loop"...
# Remember, you should never Parse result of "ls" in any case because
# unix allows every character to be used in naming files, even if a "new line character"...
# execute "touch $'A\nFile'" to make file and ls to observe it.
# Google "why parsing output of ls considered bad" to know more.

for i in $args
do
        if [ -f "$i" ]; then
                CPATH=`readlink -f "$i"`
# readlink returns the symbolic link, -f canonicalize by every parent directory recursively.
                echo $CPATH
        fi
done

noargs=${#args}
# noargs stores total number of arguments.
if [ "$noargs" -eq "0" ]; then
        echo "No such a file exists"
fi



************************GTU5************************
#!/usr/bin/env bash
# GTU 5 - Write a script to copy the file system from two directories to a new directory
# in such a way that only the latest file is copied in case there are common files
# in both the directories.
# Code written By: Rushyang Darji
# Last Build: 24.08.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

EXIT=n
while [ $EXIT != y ]
do
        sleep 1
        echo -e "\n"
        echo -e "       1. Display PWD
        2. Long Listing
        3. Change Directory
        4. Copy Newest File.
        5. Exit
        Enter Choice: \c"
        read ch

case $ch in

1)
        clear
        pwd
;;

2)
        clear
        pwd
        ls -l
;;

3)
        echo -n "Enter Absolute Path to change directory: "
        read apath

        cd $apath

        if [ $? -eq 0 ]; then
# We can also check for availibility of directory before 'cd' command by 'test -d $apath'
# i.e. 'if [ -d $apath ]'
                clear
                echo "Working Directory Changed successfully to.."
                sleep 1
                pwd
        else
                clear
                echo "Please check your PATH."
        fi
;;

4)
        clear
        echo "Enter filenames to copy. ( * - for ALL Files, ELSE Separate files by spaces )"
        read allfiles
        if [ -f $allfiles ]; then
                echo "Enter Absolute path, where to copy these files: "
                read -e cpath
                if [ -d $cpath ]; then
                        cp -u "$allfiles" $cpath
# -u copies only when the SOURCE file is newer than the destination file or
# when the destination file is missing
                else
                        echo "There is no such a directory!"
                fi
        else
                echo "There is/are no such file(s)!"
        fi
;;

5)
        clear
        echo -n "Exiting.."
        sleep 1
        echo -n "."
        sleep 1
        echo -n "."
        clear
        exit
;;

*)
        clear
        echo "Invalid Choice"
;;

esac
done


***********************GTU6************************
#!/usr/bin/env bash
# Code written By: Rushyang Darji
# Last Build: 09.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

while true; do
        read -e -p "Enter first Directory's Absolute path: " path1 || exit
        [[ -d $path1 ]] && break
                echo "Invalid path, Try Again!"
done

while true; do
        read -e -p "Enter second Directory's Absolute path: " path2 || exit
        [[ -d $path2 ]] && break
                echo "Invalid path, Try Again!"
done
```

```
while true; do
	read -e -p "Enter Third Directory's Path, to copy files in case of exact match: " path3 || exit
	[[ -d $path3 ]] && break
		echo "Invalid Path, Try, Again!"
done

temp=$(mktemp)
for i in $path1/*
do
	if [ -f "$i" ]; then
	for j in $path2/*
	do
		if [ -f "$j" ]; then
		base1=`basename "$i"`
		base2=`basename "$j"`
		if [ "$base1" = "$base2" ]; then
			diff "$i" "$j" > $temp

			size=`ls -s $temp | awk '{print $1}'`
			if [ "$size" -eq "0" ]; then
				echo "File: \"$base1\" was found same in both directories."
				cp "$i" "$path3"
				echo "Copied to \"$path3\" successfully!"
			fi
		fi
		fi
	done
	fi
done

rm $temp

: << --
	OUTPUT (Attempt 1)
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU6-2.sh
Enter first Directory's Absolute path: /home/rushyang/sldjf
Invalid path, Try Again!
Enter first Directory's Absolute path: /home/rushyang/Experiments/1510/mydir1/
Enter second Directory's Absolute path: /home/rushyang/lsajlj
Invalid path, Try Again!
Enter second Directory's Absolute path: /home/rushyang/Experiments/1510/mydir2/
Enter Third Directory's Path, to copy files in case of exact match: /home/rushyang/Experiments/1510/mydir3/
File: "1" was found same in both directories.
Copied to "/home/rushyang/Experiments/1510/mydir3/" successfully!
File: "new file" was found same in both directories.
Copied to "/home/rushyang/Experiments/1510/mydir3/" successfully!


	OUTPUT (Attempt 2)
rushyang@Maverick_Meerkat: GTU-MCA $ bash GTU6-2.sh
Enter first Directory's Absolute path: /home/rushyang/Experiments/1510/mydir1/
Enter second Directory's Absolute path: /home/rushyang/Experiments/1510/mydir2/
Enter Third Directory's Path, to copy files in case of exact match:
/home/rushyang/Experiments/1510/mydir3/
File: "1" was found same in both directories.
Copied to "/home/rushyang/Experiments/1510/mydir3/" successfully!
File: "new file" was found same in both directories.
Copied to "/home/rushyang/Experiments/1510/mydir3/" successfully!
--
########################   OR    ########################

#!/usr/bin/env bash
# Code written By: Rushyang Darji
# Last Build: 02.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
```

```
: << --
GTU 6: Write a script to compare identically named files in two different directories
and if they are same, copy
one of them in a third directory

Code Developed By: Rushyang Y Darji (rushyang@yahoo.co.in)
Init Build: 04.08.2010
Last Build: 15.10.2010
v1.6
--

clear
temp=$(mktemp tmp.XXXXXXXX)
#mktemp makes temporary file into /temp directory with r+w permissions only for creator.
echo -n "Enter 1st Directory: "
read dir1
PATH1="$(pwd)/$dir1"    #Converted whole path of dir1 into PATH1 variable
echo "PATH1=$PATH1"
main=$(pwd)
if test -d $PATH1; then #Test condition to make sure dir1 is a directory.
        echo -n "Enter 2nd Directory: "
        read dir2
        PATH2="$(pwd)/$dir2"    #Same as Line 14
        echo "PATH2=$PATH2"
        if test -d $PATH2; then
                cd $PATH1       #Secured for using for i loop.
                for i in *
                do
                        cd $PATH2       #Secured for using for j loop.
                        for j in *
                        do
                                if test "$i" == "$j"; then
                                        cd $main                #Back to $(basedir)
                                        cmp "$PATH1/$i" "$PATH2/$j" > $temp
                                        # cmp checks byte by byte.. can be little slower than 'diff'
                                        size=`ls -s $temp | awk '{print $1}'`
# '-s' for listing size, and 'awk' for fetching size of $temp
                                        if test "$size" == "0"; then
# if size of temporary file is 'ZERO', then both files are exactly same.
                                                echo "File: \"$i\" was found same in both directories."
                                                echo -n "Enter Directory name (must be in current working directory) to cop
                                                read dir3
                                                PATH3="$(pwd)/$dir3"    #Same as Line 14
                                                if test -d $PATH3; then
                                                        cp -i "$PATH1/$i" "$PATH3"
# "$PATH2/$j" instead of "$PATH1/$i" will also do.
                                                else
                                                        echo "There is no directory named $dir3 in $pwd"        #Line 41
                                                fi
                                        fi

                                fi
                        done
                done
        else
                echo "Invalid Directory Name for dir2." #Error Message of missing dir2
        fi
else
        echo "Invalid Directory Name for dir1." #Error Message of missing dir3
fi

rm $temp

: << --
rushyang@Maverick_Meerkat: 1510 $ ls mydir1/
1  2  new file
rushyang@Maverick_Meerkat: 1510 $ ls mydir2/
```

```
1  new file  T
rushyang@Maverick_Meerkat: 1510 $ ls mydir3/

                OUTPUT (Attempt 1)
Enter 1st Directory: asdf
PATH1=/home/rushyang/Experiments/1510/asdf
Invalid Directory Name for dir1.

                OUTPUT (Attempt 2)
Enter 1st Directory: mydir1
PATH1=/home/rushyang/Experiments/1510/mydir1
Enter 2nd Directory: mydir2
PATH2=/home/rushyang/Experiments/1510/mydir2
File: "1" was found same in both directories.
Enter Directory name (must be in current working directory) to copy it: mydir3
File: "new file" was found same in both directories.
Enter Directory name (must be in current working directory) to copy it: mydir3

                OUTPUT (Attempt 3)
Enter 1st Directory: mydir1
PATH1=/home/rushyang/Experiments/1510/mydir1
Enter 2nd Directory: mydir2
PATH2=/home/rushyang/Experiments/1510/mydir2
File: "1" was found same in both directories.
Enter Directory name (must be in current working directory) to copy it: mydir3
cp: overwrite `/home/rushyang/Experiments/1510/mydir3/1'? y
File: "new file" was found same in both directories.
Enter Directory name (must be in current working directory) to copy it: mydir3
--

************************GTU7************************
#!/usr/bin/env bash
# Code Developed By: Rushyang Darji
# Last Build: 14.09.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
while true; do
        read -e -p "Enter the Absolute Path: " path || exit
        [[ -d $path ]] && break
        echo "Invalid Directory!"
done


args=`find "$path" -empty -print0 | xargs -0`
for i in $args
do
        if [ -f "$i" ]; then
                rm -i "$i"
        fi
done

************************GTU8************************
#!/usr/bin/env bash
# Write a script to display the name of those files (in the given directory)
# which are having multiple links.
# Developed By: Rushyang Darji
# Init Build: 15.10.2010
# Last Build: 15.10.2010
# v1.0
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

echo "Enter Absolute path of directory"
read path

if test -d $path; then
        cd $path
        for i in *
        do
                for j in *
                do
                        if test "$i" != "$j"; then
                                if test "$i" -ef "$j"; then
                                        echo "$i" >> $$.temp
                                fi
                        fi
                done
        done
        cat $$.temp | uniq
        rm $$.temp
        cd -
else
        echo "Check your path."
fi

###############   OR   ###############

#!/usr/bin/env bash
# Write a script to display the name of those files (in the given directory)
# which are having multiple links.
# Developed By: Rushyang Darji
# Init Build: 15.10.2010
# Last Build: 15.10.2010
# v1.0
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.

temp=$(mktemp tmp.XXXXXXXXXX)
while true; do
        read -e -p "Ente the Absolute Path: " path || exit
        [[ -d $path ]] && break
        echo "Invalid Directory!"
done

for i in $path/*
do
        for j in $path/*
        do
                base1=`basename $i`
                base2=`basename $j`
                if test "$base1" != "$base2"; then
                        if test "$base1" -ef "$base2"; then
                                echo "$base1" >> $temp
                        fi
                fi
        done
done

cat $temp | uniq
rm $temp

************************GTU9************************
#!/usr/bin/env bash
# GTU 9
# Code Developed by: Rushyang Darji
# Last Updated: 14.10.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
# read -e enables Tab Completion which is way too easier than entering whole absolute path.
read -e -p "Enter Absolute Path: " path || exit
temp=$(mktemp)
if test -d $path; then

        for i in $path/*
        do
                if test -x "$i"; then
                        echo "$i" >> "$temp"
                fi
        done
else
        echo "Invalid Directory"
fi

cat $temp
rm $temp
************************GTU10************************
#!/usr/bin/env bash
# GTU10: Write a script to display the date, time and a welcome message
# (like Good Morning should be displayed with â€œa.m.â€□ or â€œp.m.â€□ and not in 24 hours notation.
# Code written By: Rushyang Darji
# Last Updated: 19.08.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.


msg2=`date +%H`
echo "Welcome $USERNAME!"
time=`date +"%F  %I:%M:%S %p"`
echo "Current Time is: $time"

if [ "$msg2" -ge "5" ] && [ "$msg2" -lt "12" ]; then
        echo "Good Morning..!"
elif [[ "$msg2" -ge "12" ]] && [[ "$msg2" -lt "17" ]]; then
        echo "Good Afternoon..!"
elif [[ "$msg2" -ge "21" ]] && [[ "$msg2" -lt "19" ]]; then
        echo "Good Evening..!"
else
        echo "Good Night..!"
fi
************************GTU11************************
while true; do
        read -e -p "Enter first Directory's Absolute path: " path
        [[ -d $path1 ]] && break
                echo "Invalid path, Try Again!"
done

ls -Sl $path # this will list all files in descending order...

ls -Sl $path | tac # This will list all files in ascending order....

#####or

cd $path
ls * -dplSr | grep -v '/$'
cd $OLDPATH
***********************GTU13***********************
echo "Enter the filename: "
read file2
lim=`wc -l < $file2`
echo $lim
i=2
while [ $i -le $lim ];
do
        temp=$(mktemp)
        sed -n "$i"p < $file2 > $temp
        roll=`cat $temp | awk '{print $1}'`
        name=`cat $temp | awk '{print $2}'`
        sub1=`cat $temp | awk '{print $3}'`
        sub2=`cat $temp | awk '{print $4}'`
        sub3=`cat $temp | awk '{print $5}'`
        total=`echo $sub1+$sub2+$sub3 | bc`
        per=`echo "scale=2; $total/3" | bc`
        echo "====== Marksheet for Student \"$i\" ======="
        echo -e "Roll\tName\tSubject1\tSubject2\tSubject3\t Grand Total \t Percentage"
        echo -e "$roll \t $name \t $sub1 \t\t $sub2 \t\t $sub3 \t\t $total \t\t $per"
        rm $temp
        i=$(( $i+1 ))
done
***********************GTU14***********************
#!/usr/bin/env bash
# Code written By: Rushyang Darji
# Last Updated: 13.08.2010
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
: << --
14. Write a script to make following file and directory management operations menu based:
        Display current directory
        List directory
        Make directory
        Change directory
        Copy a file
        Rename a file
        Delete a file
        Edit a file
--

while true; do


sleep 1
echo -e "\n"
echo -e "       1. Display current directory
        2. List directory
        3. Make directory
        4. Change directory
        5. Copy a file
        6. Rename a file
        7. Delete a file
        8. Edit a file
        9. Exit
        Enter your choice: \c"
read selection
clear

case $selection in

1) pwd
;;

2) ls -l
;;

3) echo -n "Enter name of directory you wanna make: "
read DIR
mkdir "$DIR"
if [ "$?" == "0" ]; then
        echo "Directory $DIR made successfully!"
        sleep 1
        ls -l
fi
;;

4) echo -n "Enter the Absolute Path to change the directory: "
read -e apath
cd $apath
        if [ $? -eq 0 ]; then
                echo "Working path changed successfully!"
                sleep 1
                pwd
        fi
;;

5) echo -n "Enter name of file: "
read filename
echo -n "Copy where? "
read -e apath
cp $filename $apath

if [ $? -eq 0 ]; then
        sleep 1
        echo "File $filename copied successfully to $apath"
fi
;;

6) echo -n "Enter old name of the file: "
read oname
echo -n "Enter new name: "
read nname
mv $oname $nname
;;

7) echo -n "Enter filename to delete: "
read fdel
if [ -f $fdel ]; then
        rm -i $fdel
fi
;;

8) echo -n "Enter filename to open it in Text Editor: "
read filename
vi $filename
;;

9)
clear
echo "============ HAVE A NICE DAY ============"
sleep 2
clear
exit
;;

*) echo "Invalid choice, Please try again!: "
;;

esac
done
**************************GTU15********************
#Write a script which reads a text file and output the following
#Count of character, words and lines.
#File in reverse.
#Frequency of particular word in the file.
#Lower case letter in place of upper case letter.

read -e -p "Enter the filename: " file1

if [[ -f $file1 ]]; then
        echo "Filename is \"$file1\". "
        echo "Count of Characters: " `wc -m < "$file1"`
        echo "Count of Words: " `wc -w < "$file1"`
        echo "Count of Lines: " `wc -l < "$file1"`

        echo "Reverse File is: "
        rev "$file1" | tac

        read -p "Enter the word which you want to count frequency of: " aword
        frq=`grep -c "$aword" < "$file1"`
        echo "Frequency of word \"$aword\" is: $frq"

        echo "Converting all lower case to upper case.."
        tr [:upper:] [:lower:] < "$file1"
fi


********************GTU16********************
#!/usr/bin/env bash
# Code written By: Rushyang Darji
# My Online Repository: http://github.com/rushyang/GTU-OS-Bash-Scriptss
: << --
This shell script will verify whether the user is logged in or not.
--

echo -e "Enter username to verify whether he is logged in or not: \c"
read myname

who | awk '{print $1}' > allusers_dummy.log
sed -n /^$myname$/p allusers_dummy.log > finalusers_dummy.log
SIZE=`ls -s finalusers_dummy.log | awk '{ print $1 }'`

if [ $SIZE -eq 0 ]; then
        echo "User is not logged in"
else
        echo "User: $myname is currently logged in"
        sleep 2
        who | sed -n /^$myname/p
fi

rm allusers_dummy.log
rm finalusers_dummy.log

# OR

#!/usr/bin/env bash
# Code written By: Rushyang Darji
# My Online Repository: http://github.com/rushyang/GTU-OS-Bash-Scripts

echo -e "Enter name of user: \c"
read myname


finaluser=`who | awk '{print $1}' | sed -n /^$myname$/p | head -n1`
if [ "$myname" = "$finaluser" ]; then
        echo "User is currently logged in."
        sleep 1
        who | sed -n /^$myname/p
else
        echo "User is not currently logged in. "
fi


*********************GTU17*********************
# Write a script to perform operations on data file.
# The file contains data for following fields.
# EMP/NO, NAME, AGE, GENDER, DESIGNATION, BASIC/SALARY
# Provide Menu Driven facility for:
# 1. Add Record
# 2. Delete Record
# 3. Modify Record
# 4. View All Record
# 5. Count Total Number of Records
# 6. Exit

datafile="/home/rushyang/Experiments/1911/datafile"
```

```
echo "Datafile is: $datafile"
cat $datafile

while true; do

read -p "
1. Add Record
2. Delete Record
3. Modify Record
4. View All Record
5. Count All Number of Records
6. Exit
Select Your Choice: " ch || exit

case $ch in
1)
        echo "Enter The Following data: "
        read -p "EMP NO: " empno
        read -p "EMP NAME: " empname
        read -p "AGE: " age
        read -p "GENDER: " gen
        read -p "DESIGNATION: " des
        read -p "BASIC SALARY: " basic
        echo -e "$empno\t$empname\t\t$age\t$gen\t$des\t\t$basic\n" >> $datafile
        tmp=$(mktemp tmp.XXXX)
        awk NF < "$datafile" > "$tmp"
        mv "$tmp" "$datafile"
;;
2)
        temp=$(mktemp tmp.XXXXX)
        read -p "Enter the employee id, which you want to delete: " empid
        grep -v "^$empid" < $datafile > $temp
        mv $temp $datafile
;;
3)
        temp=$(mktemp tmp.XXXXX)
        temp2=$(mktemp tmp.XXXXX)
        temp3=$(mktemp tmp.XXXXX)
        read -p "Enter the employee ID, you want to edit" empid
        grep -v "$empid" < "$datafile" > $temp
        grep "$empid" < "$datafile" > "$temp2"
        read -p "Enter Field NO to edit: " fno
        if test "$fno" = 1; then
                olddata=`cat "$temp2" | awk '{print $1}'`
                read -p "Enter new Empid" newempid
                sed s/$oldata/$newempid/g < "$temp2" > $temp3
        elif test "$fno" = 2; then
                olddata=`cat "$temp2" | awk '{print $2}'`
                read -p "Enter new name" newname
                sed s/$oldata/$newname/g < "$temp2" > $temp3
        elif test "$fno" = 3; then
                olddata=`cat "$temp2" | awk '{print $3}'`
                read -p "Enter new age: " newage
                sed s/$olddata/$newage/g < "$temp2" > $temp3
        elif test "$fno" = 4; then
                olddata=`cat "$temp2" | awk '{print $4}'`
                read -p "Enter Gender: " newgen
                sed s/$oldata/$newgen/g < "$temp2" > $temp3
        elif test "$fno" = 5; then
                olddata=`cat "$temp2" | awk '{print $5}'`
                read -p "Enter new Designation: " newdes
                sed s/$oldata/$newdes/g < "$temp2" > $temp3
        elif test "$fno" = 6; then
                olddata=`cat "$temp2" | awk '{print $6}'`
                read -p "Enter new salaray: " newsal
                sed s/$oldata/$newsal/g < "$temp2" > $temp3
        fi
        mv "$temp3" "$temp2"
        cat "$temp2" >> $temp
        cat "$temp" | uniq > "$datafile"
        rm "$temp"
;;
4)
        echo "Datafile is: $datafile"
        cat $datafile
;;
5)
        temp=$(mktemp tmp.XXXXX)
        awk NF < $datafile > $temp
        mv $temp $datafile
        lineno=`wc -l $datafile | awk '{print $1}'`
        echo "Total Number of data in the file: $(( $lineno-1 ))"
;;
6)
        exit
;;
*)
        echo "Invalid Choice."
;;
esac
done
******************GTU18*******************
#Write A Script To Perform Following String Operations Using Menu:
#COMPARE TWO STRINGS.
#JOIN TWO STRINGS.
#FIND THE LENGTH OF A GIVEN STRING.
#OCCURRENCE OF CHARACTER AND WORDS
#REVERSE THE STRING.

read -p "Enter String 1: " str1
read -p "Enter String 2: " str2

while true; do
read -p "        1. Compare Two Strings
        2. Join Two String
        3. Find the Length of a given string
        4. Calcucalte the Occurances of a Characters and words
        5. Reverse The String
        6. Exit
        Enter your Choice: " ch
case $ch in
1)
        if test "$str1" = "$str2"; then
                echo "Both Strings are same"
        else
                echo "Both Strings are no same"
        fi
;;
2)
        echo "$str1$str2"
;;
3)
        echo "Length of a string 1: ${#str1}"
        echo "Length of a string 2: ${#str2}"
;;
4)
        read -p "Enter the character to calculate the occurance: " char
        read -p "Enter the word to calculate the occurance: " word

        filetemp1=$(mktemp)
        echo "$str1" > $filetemp1
        filetemp2=$(mktemp)
        echo "$str2" > $filetemp2
        temp1=$(mktemp)
        temp2=$(mktemp)
        grep -o "$char" < $filetemp1 > $temp1
        echo "For string 1: "
        echo "Occurace of a character: " `wc -l < $temp1 | awk '{print $1}'`
        grep -o "$word" < $filetemp1 > $temp2
        echo "Occurance of Word: " `wc -l < $temp2 | awk '{print $1}'`
        echo "For string 2: "
        grep -o "$char" < $filetemp2 > $temp1
        echo "Occurace of a character: " `wc -l < $temp1 | awk '{print $1}'`
        grep -o "$word" < $filetemp2 > $temp2
        echo "Occurance of Word: " `wc -l < $temp2 | awk '{print $1}'`
        rm $filetemp1 $filetemp2 $temp1 $temp2
;;
5)
        echo "Reverse strings are:"
        echo "$str1" | rev
        echo "$str2" | rev
;;
6)
        exit
;;
*)
        echo "Invalid Choice.. Try again.."
;;
esac
done
************GTU19************
# Write a script to calculate gross salary for any number of employees
# Gross Salary =Basic + HRA + DA.
# HRA=10% and DA= 15%.

read -p "How many employees data, you want to enter? " no
i=1

while [ $i -le "$no" ];
do
        read -p "Enter basic salary of Employee $i: " basic
        HRA=`echo "scale=2; $basic*15/100" | bc`
        DA=`echo "scale=2; $basic/10" | bc`

        echo "Gross Salary of employee $i: " `echo "$basic+$HRA+$DA" | bc`
        # i=$(( $i+1 ))
        i=`expr $i + 1`
done
***************GTU24***************
#!/usr/bin/env bash
# GTU24: Write a script to display the last modified file.
# Code written By: Rushyang Darji
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts" for regular updates and more scripts.
# Final Build: 19.10.2010

while true;
do
  # -e enables readline, which means you can use tab-completion. & -p prints whatever's written in "" before taking the pat
  # The ''|| exit'' makes the script exit if read returns false, which it
  # will if the user hits Ctrl+C amongst other.
        read -e -p "Enter Directory: " path || exit
  # if path contains an existing directory, break out of this infinite loop.
        [[ -d $path ]] && break
        echo "Invalid Path, Try Again!"
done

cd $path        # cd $path is inevitable because of * in use of ls.
ls * -dpltr | grep -v '/$' | tail -n1
# Here, observe '*' after ls. You must specify a wildcard pattern for indicating all files first.
# This is because the -d option specifies that only directory names should listed.
# Moreover, -p puts an indicator at the end of "directories", which will be stripped by grep inverse.
# Once we have neglected directories, we can list(-l) "ONLY FILES" from current working directory
# sorted by it's modified time (-t) in reverse order (-r). The most last one will be fetched by tail.

cd $OLDPWD
# OLDPWD is the env var, which always remembers our "PREVIOUS WORKING DIRECTORY".
# Enter `env | grep OLDPWD` to see it.
# 'cd -' will also lead us into last working directory.
# But then also we don't need to print it on Terminal while executing it.


***************GTU25***************
#!/usr/bin/env bash
# Code written By: Rushyang Darji
# Visit My Online Repository: "http://github.com/rushyang/GTU-OS-Bash-Scripts"
# for regular updates and more scripts.
# Final Build: 02.10.2010

while true; do

        read -e -p "Enter Directory: " path || exit
        [[ -d $path ]] && break
        echo "Invalid path! Try Again!"
done
path=${path%/}
myargs=`grep -l -e "printf" -e "fprintf" $path/*.c | xargs`

if [ $? -gt 1 ]; then           # grep exits with status 1 when no matches were found.
        echo -n "No Matches were found. "  && exit
fi
temp=$(mktemp tmp.XXXXXXXXX)
for i in $myargs                # Here, grep has the exit status 0.
do
        echo "Do you want to add '#include <stdio.h>' to $i?"
        read S
        case $S in
        Y|y|YES|Yes|yes|yeah)
                 sed '1i\
#include<stdio.h>' "$i" > "$temp"
# i for insertion, 1 for 1st line. $i is the file to insert. and all output will be redirected to $temp

        mv "$temp" "$i"                 # renaming $temp by over writing to $i

        ;;

        n|N|NO|no|No|nope)
                echo "Alright! Next.."
                shift
        ;;

        *)
                echo "Invalid input."
        ;;
        esac

done

if [ -z $myargs ]; then
        echo "No Matches were found. Try another Directory"
else
        clear
        head -n5 $path/*.c | less
fi

rm $temp


***************GTU26***************
```

```
# !/usr/bin/env bash
# 26. Interactive - non-interactive shell script to prompt and delete c files within the given or predefined current direct
#
# Code Developed By: Rushyang Darji
# Init. Build: 06.08.2010
# Last Build: 19.10.2010

N=$#
ext=c
if test "$N" -eq "0"; then
        while true; do  # Same inifinite loop as we used in GTU24
                read -e -p "Enter Path: " path || exit
                [[ -d $path ]] && break
                echo "Invalid Path, Try Again!"
        done

        path=${path%/}

# Removes last / from the end of the path. Though, it's not compulsion to do so because
# /foo/bar and /foo//////bar is considered exactly the same!

        for i in $path/*.C
        do
                if [ "$i" != $path/'*.C' ]; then

# If there is no match, Value of i will be ''$path/*.C''. & That's why there is no need to rename.
                        mv "$i" "${i/.C/}".c
# Renames every .C files to .c, so that we can use it afterwards in same loop.
                        clear
                fi
        done

        for i in $path/*."$ext"
        do
                if [ "$i" != "$path"/'*.c' ]; then
# If there is no ".C FILE" exist in that directory, it will switch to else.
                clear
                echo "File is $i"
                head -n10 "$i" | nl
# head for displaying First 10 lines, nl for numbering them on terminal.
                sleep 1                 # Halt for 1 second
                rm -i "$i"              # -i for interactive prompt.
# Remember, "" around $i is super necessary! Because except it, you'll get an error with filenames containing spaces.
                else
                        echo "There are no matching \"C\" files to Prompt in this directory."
                        sleep 2
                        clear=no
                fi
        done

        if test "$clear" != "no"; then  # If clear=no then there are no C Files to display.
                clear
                echo "Remaining C files in the Directory..."
                ls -1 $path/*.c                 # 1 result per line (-1)
        fi
else    # Else part contains, where user passes the name of C files, which should exist in the current working directory as
        for i in $path/*.C
        do
                if [ "$i" != $path/'*.C' ]; then
# if There are no matches ie if there is no C file in given dir, 'i' will be ''$path/*.C''
                mv "$i" "${i/.C/}".c    # Renames every .C files to .c
                clear
                fi
        done

        for i in $*
# When filenames are passed as parameters.
```

```
        do
                clear
                i="${i/.c/}"
# Removes an extension from file variable 'i' Only in the case of extension is also passed within the filename parameter.
                i="$(pwd)/$i.c"         # Makes i the complete path of a file, including extension..
# Last two lines are necessary because user, may and may not enter filename including extension.
                if [ -f "$i" ]; then    # Checks for the existence of given filename, into pwd
                        echo "File name is $i"
                        head -n10 "$i" | nl
                        sleep 1
                        rm -i "$i"
                else
# Error for non-Existent files.
                        echo "There is no such a file with name: \"$i\" in current working directoy"
                        sleep 3
                fi
        done
        clear
        echo "Remaining C files in the Directory..."
        ls -1 *.c
        sleep 1
fi


#!/usr/bin/env bash
#***************GTU27***************
# 27. Write a script that deletes all leading and trailing spaces in all lines in a file.
# Also remove blank lines
# from a file. Locate lines containing only printf but not fprintf.
# Rushyang Darji
# Init Build: 29.11.2010
# Last Build: 29.11.2010

while true; do
        read -e -p "Enter path of a file: " filep || exit
        [[ -f "$filep" ]] && break
        echo "This is not a valid file."
done

temp=$(mktemp temp.XXXXX)
echo "After removing spaces... saved in $temp"
# Or, sed -e '/^$/d' -e '/^[<spc><tab>]*$/d' < filep > $temp
awk NF "$filep" > $temp
cat $temp

echo "Locating lines containing only printf but not fprintf.."
grep -ve "fprintf" < wrongfile | grep -e "printf"

rm -i $temp
```