1. Write a shell script to calculate simple interest.

echo "Enter Amount:"

read p

echo "Enter Time:"

read t

echo "Enter ROI:"

read r

i=` expr $p \\* $t \\* $r `

i=` expr $i / 100 `

echo "Simple Interest is: $i"

2. Write a shell script to calculate salary from given basic.

Salary = basic + dp + da +hra +ma –pf

basic – to be taken as input

dp - 50 % of basic

da - 35 % of (basic + dp)

hra - 8 % of (basic + dp)

ma - 3 % of (basic + dp)

pf - 10% of (basic + dp)

echo "enter the basic salary:"

read bsal

if [ $bsal –ne 0 ]

then

dp= $((bsal/100)\*50)

da=$((bsal+dp)/100)\*35)

hra=$((bsal+dp)/100)\*8)

ma=$((bsal+dp)/100)\*3)

pf=$((bsal+dp)/100)\*10)

sal=$((bsal+hra+da+ma+dp)-pf)

echo "The gross salary : $sal"

fi

3. Write a shell script to calculate the average of a set of N number.

# Total numbers

n=5

# copying the value of n

m=$n

# initialized sum by 0

sum=0

# array initialized with

# some numbers

array=(1 2 3 4 5)

# loop until n is greater

# than 0

while [ $n -gt 0 ]

do

    # copy element in a

    # temp variable

    num=${array[`expr $n - 1`]}

    # add them to sum

    sum=`expr $sum + $num`

    # decrement count of n

    n=`expr $n - 1`

done

# displaying the average

# by piping with bc command

# bc is bash calculator

# command

avg=`echo "$sum / $m" | bc -l`

printf '%0.3f' "$avg"

4. Write a Unix shell script to find the sum of number to given number.

e.g. if entered number is 5 then 1+2+3+4+5

# Total numbers

n=5

# copying the value of n

m=$n

# initialized sum by 0

sum=0

# array initialized with

# some numbers

array=(1 2 3 4 5)

# loop until n is greater

# than 0

while [ $n -gt 0 ]

do

    # copy element in a

    # temp variable

    num=${array[`expr $n - 1`]}

    # add them to sum

    sum=`expr $sum + $num`

    # decrement count of n

    n=`expr $n - 1`

printf '%0.3f' "$sum"

done

5. Write a shell script to perform like calculator. It should ask for the number and operand from the user.

# !/bin/bash

# Take user Input

echo "Enter Two numbers : "

read a

read b

# Input type of operation

echo "Enter Choice :"

echo "1. Addition"

echo "2. Subtraction"

echo "3. Multiplication"

echo "4. Division"

read ch

# Switch Case to perform

# calulator operations

case $ch in

  1)res=`echo $a + $b | bc`

  ;;

  2)res=`echo $a - $b | bc`

  ;;

  3)res=`echo $a \\* $b | bc`

  ;;

  4)res=`echo "scale=2; $a / $b" | bc`

  ;;

esac

echo "Result : $res"

**Assignment-2**

1. Write a shell script to calculate the area of rectangle. It should take the value from the command line.

echo "Input length of a rectangle"

read length

echo "Input breadth of a rectangle "

read breadth

echo "Input radius of a circle "

read radius

area=`echo $length \\* $breadth | bc`

perimeter=` echo 2 \\* $length \\* $breadth | bc`

echo "Area of the rectangle : $area"

echo "Perimeter of the rectangle : $perimeter"

2. Write a shell script to take two numbers from command line and show result of dividing small number with bigger number. Also note that it should not accept zero or negative number. If user enter zero or negative number then it should prompt to input correct number after displaying proper message.

echo "Read No 1"

read one

echo " Read No 2"

read two

while [$1, $2 –eq 0 ]

echo “enter proper number”

do

if $(one –lt two )

res= $(two/one)

then

res =$(one/two)

echo "Divided Result : $perimeter"

fi

3. Write a Unix Shell Script which prints the following

a. Current home directory.

b. Current user name.

c. The message "Today is :" with current date in MM/dd/yy format

d. The message "No of users logged in :" with total no of current logged in users

e. The message "Terminal :” With you own terminal number

echo "The current working directory: $PWD"

echo "The previous current working directory: $OPLDPWD"

\_cwd="$PWD"

## use pwd command ##

\_mydir="$(pwd)"

## or ##

\_mydir="`pwd`"

echo "My working dir: $\_mydir"

u="$USER"

echo "User name $u"

date +'FORMAT'

### mm/dd/yyyy ###

Echo “today Is : $(date +'%m/%d/%Y')”

echo "Currently logged on users:"

$who

echo "Input length of a rectangle"

read length

echo "Input breadth of a rectangle "

read breadth

echo "Input radius of a circle "

read radius

area=`echo $length \\* $breadth | bc`

perimeter=` echo 2 \\* $length \\* $breadth | bc`

echo "Area of the rectangle : $area"

echo "Perimeter of the rectangle : $perimeter"

4. Write a shell script to create a command line calculator.

e.g. input : mycal 5 + 5 Result : 10 , input : mycal 5 / 5

$ bc <<<"236-192"

44

$ bc <<<"1+1"

2

5. Write a shell script that that takes as command line input a number N and a Word. it then prints the word n times., one word per line

# display all the arguments using for loop

if [ $# -gt 0 && $# -lt 2 ]

then

n=$1

word=$2

echo "List of arguments:"

for n in $@

do

echo "$word"

done

fi

**Assignment-3**

1. Write a shell script to rename file having extension sh to exe.

#!/bin/sh

#Save the file as multimove.sh

IFS=$'\n'

if [ -z "$1" ] || [ -z "$2" ]

then

echo "Usage: multimove oldExtension newExtension"

exit -1

fi

# Loop through all the files in the current directory

# having oldExtension and change it to newExtension

for oldFile in $(ls -1 \*.${1})

do

# get the filename by stripping off the oldExtension

filename=`basename "${oldFile}" .${1}`

# determine the new filename by adding the newExtension

# to the filename

newFile="${filename}.${2}"

# tell the user what is happening

echo "Changing Extension \"$oldFile\" --> \"$newFile\" ."

mv "$oldFile" "$newFile"

done

2. Write a Unix shell script to accept 10 number and tell how many are +tive, - tive and zero. Also display the in ascending order.

IFS=' ' read -ra arr -p "Enter numbers: "

Enter numbers: 4 -1 2 66 10

sort -nc <(printf "%s\n" "${arr[@]}")

3. Write a shell script to examine all the number from 1 to 999 and display all those number whose sum of cube of the digit is equal to the number. e.g. 371 = 3\*3\*3+7\*7\*7+1\*1\*1

#!/bin/bash

echo "Enter A Number"

read [n -gt 0 -lt 999]

arm=0

temp=$n

while [ $n -ne 0 ]

do

r=$(expr $n % 10)

arm=$(expr $arm + $r \\* $r \\* $r)

n=$(expr $n / 10)

done

echo $arm

if [ $arm -eq $temp ]

then

echo "Armstrong"

else

echo "Not Armstrong"

fi

4. Write a shell script to display Date in different format along with Time.

a=$(printf '%s\n' "$Prev\_date" | awk '{

printf "%04d-%02d-%02d\n", $6, \

(index("JanFebMarAprMayJunJulAugSepOctNovDec",$2)+2)/3,$3}')

$ date +%c -d "$d"

Tue 31 Dec 2013 01:13:06 PM CET

$ date +'Today is %F' -d "$d"

Today is 2013-12-31

$ date +%c -d "$d"

Tue 31 Dec 2013 01:13:06 PM CET

$ date +'Today is %A' -d "$d"

Today is Tuesday

$ date +'Today is %F' -d "$d"

Today is 2013-12-31

5. Write a shell script to accept fine number and display max and min value. If any two or three or four values are same then display proper message.

# !/bin/bash

clear

echo –n “Enter how many numbers : “

read n

echo –n “Enter integer value : “

read num

max=$num

min=$num

for ((i=1; i<n; i++))

do

echo –n “Enter integer value : “

read newnum

if [ $newnum –gt $max ] ; then

max=$newnum

fi

if [ $newnum –lt $min ] ; then

min=$newnum

fi

done

echo –n “The maximum number is : $max”

echo

echo –n “The minimum number is : $min”

sort n | uniq –d

**Assignment-4**

1. Write a shell script to find out how many file and directory are their in current directory. Also list the file and directory name separately.

#!/bin/sh

if [ -d "$@" ]

then

find "$@" -type f | ls -l "$@" | wc -l | echo "Number of files is $@"

find "$@" -type d | ls -l "$@" | wc -l | echo "Number of directories is $@"

fi

2. Write a shell script to create a menu driven program for adding, deletion or finding a record in a database. Database should have the field like rollno, name, semester and marks of three subjects. Last option of the menu should be to exit the menu.

clear

i=\"y\"

echo \"Enter name of database \"

read db

while [ $i = \"y\" ]

do

clear

echo \"1.View the Data Base \"

echo \"2.View Specific Records \"

echo \"3.Add Records \"

echo \"4.Delete Records \"

echo \"5.Exit \"

echo \"Enter your choice \"

read ch

case $ch in

1)cat $db;;

2)echo \"Enter RollNumber \"

read id

grep -i \"$id\" $db;;

3)echo \"Enter new std id \"

iread tid

echo \"Enter new name:\"

read tnm

echo \"Enter designation \"

read des

echo \"Enter college name\"

read college

echo \"$tid $tnm $des $college\">>$db;;

echo \"Enter marks of 3 sub \"

read (m1,m2,m3)

$marks = $(m1+m2+m3)

4)echo \"Enter Id\"

read id

# set -a

# sed \'/$id/d\' $db>dbs1

grep -v \"$id\" $db >dbs1

echo \"Record is deleted\"

cat dbs1;;

5)exit;;

\*)echo \"Invalid choice \";;

esac

echo \"Do u want to continue ?\"

read i

if [ $i != \"y\" ]

then

exit

fi

done

3. Write a shell script to accept the employee name from the user and display appropriate message. Assume employee is working on a single project. Employee project details are stored in the file proj.dat and employee data are stored in

emp.dat

proj.dat emp.dat

emp\_no int emp\_no int

proj\_no int proj\_no int

dayswork int name char

duration int

if message

duration = dayswork <name>, Today is last day of project.

duration < dayswork <name>, you are delaying the project.

daysremaing <= 30 <name>, hurry up only <days> are remaining.

daysremaing > 30 <name>, you have still <days> to finish the project.

echo -e "Enter Name of the employee:\c"

read name

ecd=0

rm=0

if ecd=`grep "$name" project.dat | cut -d "|" -f 1`

then

dw=`grep "$ecd" struct.dat | cut -d "|" -f 3`

du=`grep "$ecd" project.dat | cut -d "|" -f 4`

else

echo -e "===Name Not Found===\n"

fi

if [ $dw -eq $du ] ; then

echo "$name,Today is the last day to finished your project"

else

rm=`expr $du - $dw` if [ $rm -le 30 ] ; then

echo "$name,Hurry! only $rm days are remaining to finish your project"

else

echo "$name,schedule yourself you have still $rm days to finish the job"

fi

fi

4. Write a shell scripts to count number of vowels in file ignoring the case.

file=$1  
file\_contents=$(<$file) # echo "$file\_contents" arr=($file\_contents) # echo ${#arr[@]} #print the array count=0 for wo

======================================================

read word   
lengthofword=`echo $word | wc -c`   
cntr=1   
no\_of\_vowels=0   
while($cntr -le $lengthofword)   
do   
{   
ltr=`cat $word | cut -c$cntr`   
if [ $ltr = "a" -o $ltr = "e" .... $ltr = "u" ]   
then   
no\_of\_vowels=`expr $no\_of\_vowels + 1 `   
cntr=`expr $cntr + 1 `   
}   
done

5. Write a shell program which will work like wc.

echo Enter the filename

read file

w=`cat $file | wc -w`

c=`cat $file | wc -c`

l=`grep -c "." $file`

echo Number of characters in $file is $c

echo Number of words in $file is $w

echo Number of lines in $file is $l

**Assignment-5**

1. Write a unix shell to add records to a file called item.dat The fields being itemcode, qty, sold and rate

item\_code to be generated

qty\_sold should be greater than 0

rate between 100 to 10000

sudo tee -a item.dat > /dev/null <<EOT

n=$RANDOM

echo "Item code: $n..."

m=$RANDOM

echo "qty: $m...

rate=echo $((RANDOM%10000+100))

EOT

2. Write a Unix shell script which accepts a choice from the user and execute the corresponding option using case structure for above file structure. The format of the menu is

+ -> Addtion of record

! -> Modification of records

- -> Deletion of records

s -> Quit

echo 1. Addition of record

echo 2. Modify a record

echo 3. Deletion of record

echo 4. Quit

echo -e "Enter your choice: \c"

read ch

cat mat.dat

case "$ch" in

A) mcode=`tail -1 mat.dat | cut -d"|" -f1`

echo -e "\nEnter the quntity of sold item: \c"

read qty

echo -e "\nEnter the amount: \c"

read amt

(( code = mcode + 1 ))

echo "$code | $qty | $amt" >> mat.dat;;

M) echo -e "\nWhich record u want to change?: \c"

read r

echo -e "\nEnter new quantity: \c"

read nqty

echo -e "\nEnter new rate: \c"

read nrate

sed ''$r' c\ $r' | '$nqty' | '$nrate'' mat.dat >ext

cat ext > mat.dat;;

D) echo -e "Enter record no: \c"

read rn

echo -e "Are you sure to delete this record?(Y or N): \c"

read cho

if [ $cho = 'Y' ]

then

sed ''$rn' d' mat.dat >ext

fi;;

Q)esac

echo -e "ENTER BOOK NAME:"

read bname

grep "$bname" book.dat

==============================================================

clear

echo "Menu "

echo "1. COpy a File "

echo "2. Remove a file "

echo "3. Move a file"

echo "4. Quit"

echo "Enter ur Choice \c"

read Choice

case"$Choice"in

1) echo "Enter File name to copy \c"

read f1

echo "Enter FIle name \c "

read f2

if [ -f $f1 ]

then

cp $f1 $f2

else

echo "$f1 does not exist"

fi

;;

2) echo "Enter the File to be removed "

read r1

if [ -f $r1 ]

then

rm -i $r1

else

echo "$r1 file does not exist "

fi

;;

3)

echo "Enter File name to move \c"

read f1

echo "Enter destination \c "

read f2

if [ -f $f1 ]

then

if [ -d $f2 ]

then

mv $f1 $f2

fi

else

echo "$f1 does not exist"

fi

;;

4)

echo "Exit......."

exit;;

esac

3. Write a shell scripts to generate random number. Allow user to guess 6 times to get the no. Print appropriate messages after each guess.

e.g. Random generated is 10

Users enters 4 message : To small

User enters 14 then Messge to big

ctr=0

while true

do

s=`date "+%S"`

m=`date "+%M"`

let rn=s\\*m

let ctr=ctr+1

echo -e "Enter your guess \c"

read gu

echo "Random Generated Is: $rn"

if [ $gu -gt $rn ] ; then

echo "Too Big"

elif [ $gu -lt $rn ] ; then

echo "Too Small"

else

echo "Equals"

fi

if [ $ctr -eq 5 ] ; then

exit

fi

done

=========================================================

#!/bin/bash

Target= $(($RANDOM % 100))

guess=

until[[$guess –eq $target]]; do

read –p “take a gues:” guess

if [[$guess –lt $target]]; then

echo “Higher”

elif[[$guess –gt $target]]; then

echo “lower”

else

echo”you found it!”

fi

done

exit 0

4. Write a Unix Shell script that takes a login as a command line argument and reports to you when that person logs in.

# <TYBCA - SEM 5>

echo $"user name:" $LOGNAME

echo $"Shell:" $SHELL

============================================================

#!/bin/bash

clear

echo " 1 = Login"

read number

case $number in

1)

if [ -s /home/alex/Documents/PASS/password\_PASS ]; then

existpass=`cat /home/alex/Documents/PASS/password\_PASS`

echo "Type the existing pass:"

read givenpassasexisted

if [ "$existpass" = "$givenpassasexisted" ]; then

clear; echo "Success!"

else echo "Bad"

exit

fi

else echo "There is no password."

fi;;

logged=$(who | awk -v IGNORECASE=1 -v usr=$1 '{ if ($1==usr) { print $1 }exit }')

if [ -z $logged ]; then

echo "$1 is not logged on."

echo "Exit"

exit

fi

esac

5. Write shell script to accept marks of of Unix, VB.Net,PHP and calculate the average of marks and display the message as follow :

if Average then display

- is 70 or above "Distinction"

- is 60 <= 70 "First Class"

- is 50 <= 60 "Second Class"

- is 40 <= 50 "Third Class"

- otherwise "Fail"

echo "Enter the marks of Unix, VB,.Net,PHP "

read Unix VB dotNet PHP

sum1=`expr $Unix + $VB + $dotNet + $PHP `

echo "Sum of 4 subjects are: " $sum1

per=`expr $sum1 / 4`

echo " Percentage: " $per

if [ $per -ge 70 ]

then

echo "You get Distinction”

elif [ $per -ge 60 ]

then

echo “You get First class”

elif [ $per -ge 50 ]

then

echo "You get Second class"

elif [ $per -ge 40 ]

echo "You get Third class"

else

echo "You get Fail"

fi

**Assignment-6**

1. Write a script to find out String is palindrome or not.

clear

echo "Enter a string to be entered:"

read str

echo

len=`echo $str | wc -c`

len=`expr $len - 1`

i=1

j=`expr $len / 2`

while test $i -le $j

do

k=`echo $str | cut -c $i`

l=`echo $str | cut -c $len`

if test $k != $l

then

echo "String is not palindrome"

exit

fi

i=`expr $i + 1`

len=`expr $len - 1`

done

echo "String is palindrome"

1. Write a Shell script to calculate the gross salary of Rajesh whose Basic is input through keyboard. His DA is 34% of basic salary and HRA is 24% of Basic. PF is 11% of the basic.

echo -e "Enter ur basic salary \c"

read sal

if [ $sal -ge 1000 ]

then

da=`expr $sal \\* 34 / 100`

hRa=`expr $sal \\* 24 / 100`

pf=`expr $sal \\* 11 / 100`

Nsal=`expr $sal + $da + $ha`

echo "ur Basic Salary $sal "

echo "ur Dearness Allowance $da "

echo "Ur House rent $hRa "

echo "Ur PF $pf "

echo " ------------"

echo "Ur Net Salary is Rs. $Nsal "else

echo "Pls enter basic salary greater than 1000 "

fi

3. Write a shell script for accepting the following information and storing it in a file.

CD No., Movie Name, Language, Price and Date of release

# <TYBCA - SEM 5>

# <SEM - 27>

ans="y"

while [ $ans == "y" ]

do

echo "Enter the cassatte no:"

read cn

echo "Enter the Movie name:"

read mn

echo "Enter the language:"

read ln

echo "Enter the price:"

read p

echo "Enter the date or release:"

read dr

echo " $cn | $mn | $ln | $p | $dr ">>movie.dat

echo "Do you want to continue?[Y/N]"

read ans

done

# #QUE:31shell procedure to delete the specified record from

the

# library file. It should accept a book number, check

whether it is

# existing. If so, display the current details, ask for

confirmation

# and then delete it from the file.

# <TYBCA SEM 5>

clear

echo -n "ENTER NUMBER OF THE BOOK:="

read Bn

c=`grep -c "$Bn" lib1.txt`

if [ $c -gt 0 ]

then

grep "^$Bn" lib1.txt > list2

awk -F "|" '{ printf $1,$2,$3,$4,$5 }' list2

echo `grep $Bn lib1.txt`

echo "DO YOU WANT TO DELETE(Y OR N)?"

read ans

if [ $ans = 'y' -o $ans = 'Y' ]

then

echo "THE BOOK IS DELETED"

grep -v "^$Bn" lib1.txt > list1

mv list1 lib1.txt

else

echo "BOOK IS NOT DELETED"

fi

else

echo "FILE DOES NOT EXIST"

fi

1. Write a scripts which copies the content of file1 to file2 without using cp command It should check If file has a read permissions if not it should print an error message. If file2 exits then it should ask the user whether he wants to overwrite it.

ls;

echo "reading main file...";

filenames="filenames";

exec<${filenames}

while read name

do

echo "file: ${name}";

echo "reading..."

cd test1;

exec<${name}

value=0

while read line

do

#value='expr ${value} +1';

echo ${line};

done

echo "read done for ${name}";

cp ${name} ../test2;

cd ..;

echo "file ${file} moved to test2";

done

5. Write a shell scripts that delete all files in current directory with 0 byte.

echo "Enter name of directory"

read dname

echo "This Will remove $dname and all the subdirectories below it"

echo -e "Enter y/n :\c"

read ch

if [ $ch == "n" -o $ch == "N" -o -z $dname ] ; then

exit 1

fi

cd $dname

while true ; do

for i in `ls` ; do

if [ -d $i ] ; then

dir="true"

dn=$i

break;

else

dir="false"

fi

done

if [ $dir == "true" ] ; then

cd $dn

else

for i in `ls` ; do

rm $i

done

ds=`pwd`

dn=${ds##\*/}

cd ..

rmdir $dn

fi

if [ $dn == $dname ] ; then

break fi

done

**Assignment-7**

1. Write a shell script to display a directory listing as follows. Your home directory is <home directory name>

File name date time permission

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

Filename1 date time permission

Filename2 date time permission

Filename3 date time permission

………..

………..

Total no. of files : <total number>

Total no of normal file : <number>

Total no of directory : <number>

dir=$dir

for file in $dir; do

if [[ -d $file ]]; then

echo "$file is a directory"

if [[ -f $file ]]; then

echo "$file is a regular file"

fi

done

=======================================

echo "YOUR LOGIN DIRECTORY IS:" $HOME

echo "YOUR PRIMARY PROMPT IS:"

echo "NOW CURRENT MONTH IS:" `date +%B`

echo "FILES CAN BE EXECUTED AT:" $PATH

2. Write a shell to accept the details for movie LIBRARY.

movie No.

Name of the movie

Director

actor

actress

Year of release

Validations

1) Movie no. should not exist in the file.

2) 2,3 and 4th entry cannot be blank

3) year of release cannot be >= 2008

4) Year or release should be less than 1850

5) Category can be comedy, action, thriller, Suspense

The user should be able to repeatedly enter the data

ans="y"

while [ $ans == "y" ]

do

echo "Enter the cassatte no:"

read cn

echo "Enter the Movie name:"

read mn

echo "Enter the language:"

read ln

echo "Enter the price:"

read p

echo "Enter the date or release:"

read dr

echo " $cn | $mn | $ln | $p | $dr ">>movie.dat

echo "Do you want to continue?[Y/N]"

read ans

done

# #QUE:31shell procedure to delete the specified record from

the

# library file. It should accept a book number, check

whether it is

# existing. If so, display the current details, ask for

confirmation

# and then delete it from the file.

# <TYBCA SEM 5>

clear

echo -n "ENTER NUMBER OF THE BOOK:="

read Bn

c=`grep -c "$Bn" lib1.txt`

if [ $c -gt 0 ]

then

grep "^$Bn" lib1.txt > list2

awk -F "|" '{ printf $1,$2,$3,$4,$5 }' list2

echo `grep $Bn lib1.txt`

echo "DO YOU WANT TO DELETE(Y OR N)?"

read ans

if [ $ans = 'y' -o $ans = 'Y' ]

then

echo "THE BOOK IS DELETED"

grep -v "^$Bn" lib1.txt > list1

mv list1 lib1.txt

else

echo "BOOK IS NOT DELETED"

fi

else

echo "FILE DOES NOT EXIST"

fi

4. Write a shell script to menu driven program that is update and delete student records.

clear

i="y"while [ $i = "y" ]

do

echo "1.Display current dir"

echo "2.Listing the dir"

echo "3.Make a dir"

echo "4.Copy a file"

echo

i="y"while [ $i = "y" ]

do

echo "1.Display current dir"

echo "2.Listing the dir"

echo "3.Make a dir"

echo "4.Copy a file"

echo "5.Rename file"

echo "6.Delete file"

echo "7.Edit file"

echo "8.Exit"

echo "Enter your choice"

read ch

case $ch in

1)echo "Current Dir is : "

pwd;;

2)echo "Directories are"

ls;;

3)echo "Enter dir name to create"

read d

mkdir $d

echo $d" Dir is created";;

4)echo "Enter filename from copy"

read f1

echo "Enter filenm2 to be copied"

read f2

#cat $f1 > $f2

cp $f1 $f2

echo $f2" is copied from "$f1;;

5)echo "Enter file name to rename"

read f1

echo "Enter new name of file"

read f2

mv $f1 $f2

echo $f1" is renamed as "$f2;;

6)echo "Enter any filenm to be delete"

read f1

rm $f1

echo $f1" is deleted";;

7)echo "Enter any file to be editing "

read f1

vi $f1;;

8)echo "Have a nice time"

exit;;

\*)echo "Invalid choice entered";;

esac

echo "Do u want to continue ? "

read i

if [ $i != "y" ]

then

exit

fi

done

5. Write a shell script to accept 2 matrices and add them & display.

matrix[i][j] = array[i\*n + j]

|  |
| --- |
| #Bash script to add two matrix  matrix1=(1 2 3 4 5 6 7 8 9) #Matrix of size 3 by 3  matrix2=(11 12 13 14 15 16 17 18 18 19) #Matrix of size 3 by 3  rows=3  cols=3  echo "First matrix"  for((i=0; i<rows; i++))  do    for((j=0; j<cols; j++))    do      index=$((i\*cols+j))      echo -n "${matrix1[index]} "    done    echo  done    echo "Second matrix"  for((i=0; i<rows; i++))  do    for((j=0; j<cols; j++))    do      index=$((i\*cols+j))      echo -n "${matrix2[index]} "    done    echo  done    k=0  matrix3=()  for((i=0; i<rows; i++))  do    for((j=0; j<cols; j++))    do      index=$((i\*cols+j))      matrix3[k]=$((${matrix1[index]} + ${matrix2[index]}))      k=$((k+1))    done  done  echo "Addition of two matrix"  for((i=0; i<rows; i++))  do    for((j=0; j<cols; j++))    do      index=$((i\*cols+j))      echo -n "${matrix3[index]} "    done    echo  done |

**Assignment-8**

1. Write a shell script to list all the files of the current directory having read and write permission to the user.

for File in \*

doif [ -r $File -a -w $File -a -x $File ]

then

echo $File

fi

done

1. Write a shell script to checks if name given is file or directory and if it is file then it should display content and if it is a directory then it should display the list.

#!/bin/bash

PASSED=$1

if [ -d "${PASSED}" ] ; then

echo "$PASSED is a directory";

else

if [ -f "${PASSED}" ]; then

echo "${PASSED} is a file";

else

echo "${PASSED} is not valid";

exit 1

fi

fi

1. Write a shell script to accept two filenames and check if both exits. If second file exists then append the content of first file to second file.
2. Write a shell script to input a number & display in words.

echo -n "Enter number : "

read n

len=$(echo $n | wc -c)

len=$(( $len - 1 ))

echo "Your number $n in words : "

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

do

   # get one digit at a time

    digit=$(echo $n | cut -c $i)

   # use case control structure to find digit equivalent in words

    case $digit in

        0) echo -n "zero " ;;

        1) echo -n "one " ;;

        2) echo -n "two " ;;

        3) echo -n "three " ;;

        4) echo -n "four " ;;

        5) echo -n "five " ;;

        6) echo -n "six " ;;

        7) echo -n "seven " ;;

        8) echo -n "eight " ;;

        9) echo -n "nine " ;;

    esac

done

# just print new line

echo ""

5. Write a shell script to input a file name and display rights.

echo -n "Enter file name : "

read file

# find out if file has write permission or not

[ -w $file ] && W="Write = yes" || W="Write = No"

# find out if file has excute permission or not

[ -x $file ] && X="Execute = yes" || X="Execute = No"

# find out if file has read permission or not

[ -r $file ] && R="Read = yes" || R="Read = No"

echo "$file permissions"

echo "$W"

echo "$R"

echo "$X"