**SIDDHARTH RAMAKRISHNAN\_142897 UNIX ASSIGNMENT**

**LAB 01- CONNECTING TO THE UNIX SERVER**

**1.1: Connecting to the Unix Server**

**Enter your login name and password to login to the UNIX system.**

login as: user50

password: user50

**1.2: Logging out of the system**

[root@yb ]exit

**LAB 02- UNIX BASIC COMMAND**

**2:1 Executing basic commands.**

1. **To display the current working directory, the command is:**

Pwd

1. **Display the path to and name of your HOME directory.**

Pwd

1. **Display the login name using which you have logged into the system**

login as: user25

1. **Display the hidden files of your current directory.**

[user25@pace ~]$ ls -la

total 380

1. **List the names of all the files in your home directory.**

[user25@pace ~]$ pwd

/home/user25

[user25@pace ~]$ ls

a1 cfile2.bak f2 firstfile name2.sh PGM1

A1 con.sh f3 for.sh name.sh PGM2.sh

arg.sh emp file geekfile.txt num result

backup.sh err file1 j1 number.sh second

cfile1 ErrFile filegrep jumbo param.sh user.sh

cfile2 f1 file.sh marks.txt payal wfile

1. **Using the long listing format to display the files in your directory.**

[user25@pace ~]$ ls –l

1. **List the files beginning with chap followed by any number or any lower case alphabet. (Example, it should display all files whose names are like chap1, chap2, chap3 ……., chapa, ahapb, chapc,……..)**

**8. Give appropriate command to create a directory called C\_prog under your home directory. (Note: Check the directory using ls)**

mkdir C\_prog

ls

c\_prog trg1 user1 User1 user10 user11 user2

**9. Create the following directories under your home directory. (Note: Check using ls)**

mkdir newdir newdirectory

ls

c\_prog newdir newdirectory trg1 user1 User1 user10 user11 user2

**10. List the names of all the files, including the contents of the sub directories under your home directory.**

ls –r

**11. Remove the directory called newdirectory from your working directory.**

rmdir newdirectory

**12. Create a directory called temp under your home directory.**

mkdir temp

ls

**13. Remove the directory called newdir under your home directory and verify the above with the help of the directory listing command.**

rmdir newdir

ls -d \*/

ls -ld \*/

**14. Create another directory directorynew under the temp directory**

mkdir temp/directorynew

ls

cd temp

ls

**15. Change the directory to your home directory.**

cd

**16. From your home directory, change the directory to directorynew using relative and absolute path.**

cd temp/directorynew

pwd

**17. Remove the directory called c\_prog, which is in your home directory.**

rmdir C\_prog

ls

**18. Change to the directory /etc and display the files present in it.**

cd /etc

ls

**19. List the names of all the files that begin with a dot in the /usr/bin directory.**

ls -la

cat .

**20. Create a file first.unix with the following contents.**

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

cat >first.unix

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

**21. Copy the file first.unix in your home directory to first.unics.**

**(Note: checked using ls, first.unix file also should exist along with first.unics)**

cp first.unix first.unics

cat first.unics

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

cat first.unix

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

**22. List the contents of first.unix and first.unics with a single command.**

cat first.unix first.unics

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

Hi! Good Morning everybody.

Welcome to the First exercise on UNIX.

Hope you enjoy doing the assignments.

**23. Create a new directory under the temp directory.**

mkdir temp/new

**24. From your home directory, copy all the files to the directory created under the temp sub directory.**

cp -r \* temp/new

cd temp

cd new

ls

**25.Move the file first.unix to the directory temp as second.unix.**

mv first.unix temp/second.unix

ls

cd temp

ls

directorynew new second.unix

**26. Remove the file called first.unics from the home directory.**

rm first.unics

ls

**27. Change your directory to temp and issue the command rm \*. What do you observe?**

cd temp

ls

directorynew new second.unix

rm \*

rm: cannot remove `directorynew': Is a directory

rm: cannot remove `new': Is a directory

ls

directorynew new

**28. Move all files whose names end with a, c and o to the HOME directory.**

mv \*a /root/Desktop

**29. Copy all files that end with a ‘UNIX’ to the temp directory.**

cp \*.unix /home/root/temp

cd

ls

**30. Issuing a single command, remove all the files from the directory temp and the directory itself.**

rm -rf temp

ls

**31. Try commands cp and mv with invalid number of arguments and note the results.**

cp aaa ppp qqq

cp: target `qqq' is not a directory

mv aaa ppp lll

mv: target `lll' is not a directory

**32. Use the cat command to create a file friends, with the following data:**

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292 09/09/60

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

The fields should be separated by a tab.

cat >friends

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

**33. Display contents of the file friends.**

cat friends

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

**34. Copy contents of friends to newfriend without using the cp command.**

cat <friends >newfriend

cat newfriend

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

**35. Display contents of the file friends and newfriends in a single command.**

cat friends newfriend

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

Pavi 6966456 09/07/68

Sachin 2345215 08/09/67

Shankar 5546785 01/04/66

Revanth 7820022 09/07/68

Pooja 8281292

Sushma 7864563 12/12/70

Gayu 2224311 30/05/68

**36. Find all users currently working on the system and store the output in a file named as users.**

who >users

cat users

**37.Append contents of friends file to the file, users.**

cat friends >>users

cat users

**38. Display current system date and time and record your observations. How is the time displayed?**

date

Sat Dec 9 18:05:14 IST 2017

**39. Display calendar for the month and year of your birth.**

cal 9 1995

**40. Try following commands and record your observations.**

date “+ %”

date “+%m”

date “+%D”

date “+%/%Training Activity”

date “+%Training Activity”

date “+%r”

date "+ %"

%

date "+%m"

12

date "+%D"

12/09/17

date "+%/%Training Activity"

%/18:11:57raining Activity

date "+%Training Activity"

18:12:34raining Activity

date "+%r"

06:12:59 PM

**Lab 03-UNIX File System & Permissions**

**3.1: Viewing the File System and Granting/Removing Permissions**

**(Note: Create required files if doesn’t exists.)**

**1. Give the execute permission for the user for a file chap1.**

ls -l chap1

-rw-r--r-- 1 root root 0 Dec 9 18:15 chap1

chmod u+x chap1

ls -l chap1

-rwxr--r-- 1 root root 0 Dec 9 18:15 chap1

**2. Give the execute permission for user, group and others for a file add.c**

ls -l add.c

-rw-r--r-- 1 root root 0 Dec 9 18:16 add.c

chmod ugo+x add.c

ls -l add.c

-rwxr-xr-x 1 root root 0 Dec 9 18:16 add.c

**3. Remove the execute permission from user, give read permission to group and others for a file aa.c**

ls -l aa.c

-rwx--x--x 1 root root 0 Dec 9 07:54 aa.c

chmod u-x aa.c

chmod go+r aa.c

ls -l aa.c

-rw-r-xr-x 1 root root 0 Dec 9 07:54 aa.c

**4. Give execute permission for users for a.c, kk.c, nato and myfile using single command**

touch a.c kk.c nato myfile

chmod u+x a.c kk.c nato myfile

**5. Change the directory to root directory. Check the system directories, like bin, etc, usr etc.**

cd /

cd bin

cd ..

cd etc

cd ..

cd usr

**Lab 04-** **Simple and Advance Filters**

**4.1: Using Pipes and Filters:**

**1: Redirect the content of the help document ls, into a file called as lsdoc.**

ls --help >lsdoc

**2: Display the content of the lsdoc page wise.**

more lsdoc

**3: Display only the first 4 lines of the lsdoc file.**

head -4 lsdoc

Usage: ls [OPTION]... [FILE]...

List information about the FILEs (the current directory by default).

Sort entries alphabetically if none of -cftuvSUX nor --sort.

**4: Display only the last 7 lines of the file lsdoc.**

tail -7 lsdoc

--color=auto, color codes are output only if standard output is connected

to a terminal (tty). The environment variable LS\_COLORS can influence the

colors, and can be set easily by the dircolors command.

Exit status is 0 if OK, 1 if minor problems, 2 if serious trouble.

Report bugs to <bug-coreutils@gnu.org>.

**5: Remove the file lsdoc.**

rm lsdoc

**6: There will be B’day celebration from the friends file, find how many B’day** **parties will be held. If two of the friends have the B’date on the same day, then** **we will be having one party on that day.**

cut -f3 friends|sort|uniq|wc -l

**7: Display the lines starting with Ma, in the file friends.**

grep "^Ma" friends

**8: Display the lines starting with Ma, ending with i or ending with id, in the file** **friends.**

grep "^Ma[a-z]\*[i$id$]" friends

**9: Print all the files and the directory files from the current directory across all the** **sub directories, along with its path.**

**10: Print only the Directory files.**

ls -d \*/

**12: Sort the file friends in ascending order of names.**

sort friends

**13: Display the contents of the file friends in uppercase letters.**

tr "[a-z]" "[A-Z]" <friends

**14: Store the contents of your home directory in a file called dir.**

cp -r \* /home/testuser2/dir

cp: cannot copy a directory, `dir', into itself, `/home/root/dir/dir'

cd dir

ls

**15: From the above file dir, display the file permissions and the name of the file** **only.**

ls -l dir|tr -s " " |cut -d" " -f1,9|grep -v d

total

**16: From the same dir file, store only the file names in a file called files.**

ls -l dir|tr -s " " |cut -d" " -f9|grep -v d >files

cat files

**17: From the same dir file, store only the permissions of files in a file called perms.**

ls -l|tr -s " "|cut -d" " -f1 >perms

cat perms

ls -l dir|tr -s " "|cut -d" " -f1 >perms

cat perms

**18: From the same dir file, store only the file sizes in a file called sizes.**

-bash-3.2$ ls -l|tr -s " "|cut -d" " -f5 >sizes

-bash-3.2$ cat sizes

-bash-3.2$ ls -l dir|tr -s " "|cut -d" " -f5 >sizes

-bash-3.2$ cat sizes

**19: Display the file names, sizes and permissions from your directory in that** **order.**

-bash-3.2$ paste files sizes perms

**20: Display the number of users working on the system.**

who |wc -l

**21: Find out the smallest file in your directory.**

ls -l dir|tr -s " "|cut -d" " -f5,9|sort -n |head -8

**22: Display the total number of lines present in the file friends.**

-bash-3.2$ wc -l friends

**23:Create the following fixed record format files (with “|” delimiter between fields) with the structure given below, and populate them with relevant data use these files to solve following questions**

emp.lst: Empid(4),Name(18),Designation(9),Dept(10),Date of Birth(8),Salary(5)

dept.lst : Dept.Code(2),Name(10),Head of Dept’s id(4)

desig.lst: Designation Abbr.(2), Name (9)

**1.Find the record lengths of each file.**

wc -l emp.lst

4 emp.lst

wc -l dept.lst

3 dept.lst

wc -l desig.lst

3 desig.lst

**2.Display only the date of birth and salary of the last employee record.**

tail -1 emp.lst| cut -d"|" -f5,6

23/11/95|29000

**3.Extract only employee names and designations. (Use column specifications).** **Save output as cfile1.**

cut -d"|" -f2,3 emp.lst >cfile1

cat cfile1

Name(18)|Designation(9)

john|manager

jay|manager

hema|manager

**4.Extract Emp.id, dept, dob and salary. (Use field specifications). Save output as cfile2.**

cut -d"|" -f1,4,5,6 emp.lst >cfile2

cat cfile2

Empid(4)|Dept(10)|Date\_of\_birth(8)|Salary(5)

100|BI|09/06/90|50000

1001|SAP|09/06/96|30000

1002|SAP|23/11/95|29000

**5.Fix the files cfile1 and cfile2 laterally, along with the delimiter.**

paste cfile1 cfile2

Name(18)|Designation(9) Empid(4)|Dept(10)|Date\_of\_birth(8)|Salary(5)

john|manager 100|BI|09/06/90|50000

jay|manager 1001|SAP|09/06/96|30000

hema|manager 1002|SAP|23/11/95|29000

**6.Sort the emp.lst file in reverse order of Emp. Names.**

sort -t"|" -r -k2 emp.lst

Empid(4)|Name(18)|Designation(9)|Dept(10)|Date\_of\_birth(8)|Salary(5)

100|john|manager|BI|09/06/90|50000

1001|jay|manager|SAP|09/06/96|30000

1002|hema|manager|SAP|23/11/95|29000

**7.Sort the emp.lst file on the salary field, and store the result in file srtf.**

sort -t"|" -n -k6 emp.lst

Empid(4)|Name(18)|Designation(9)|Dept(10)|Date\_of\_birth(8)|Salary(5)

1002|hema|manager|SAP|23/11/95|29000

1001|jay|manager|SAP|09/06/96|30000

100|john|manager|BI|09/06/90|50000

1. **Sort the emp.lst file on designation followed by name.**

sort -t"|" -k2 emp.lst

1002|hema|manager|SAP|23/11/95|29000

1001|jay|manager|SAP|09/06/96|30000

100|john|manager|BI|09/06/90|50000

Empid(4)|Name(18)|Designation(9)|Dept(10)|Date\_of\_birth(8)|Salary(5)

-bash-3.2$ sort -t"|" -k3 -k2 emp.lst

Empid(4)|Name(18)|Designation(9)|Dept(10)|Date\_of\_birth(8)|Salary(5)

100|john|manager|BI|09/06/90|50000

1001|jay|manager|SAP|09/06/96|30000

1002|hema|manager|SAP|23/11/95|29000

**9.Sort the emp.lst file on the year of birth.**

sort -t"|" -n -k5.7 emp.lst

**10. Find out the various designations in the employee file. Eliminate duplicate listing of designations.**

cut -d"|" -f3 emp.lst | uniq

Designation(9)

manager

**11.Find the non-repeated designation in the employee file.**

cut -d"|" -f3 emp.lst | uniq -d

manager

cut -d"|" -f3 emp.lst | uniq -d |wc -l

**12.Find the number of employees with various designations in the employee file.**

cut -d"|" -f3 emp.lst | uniq -d |wc -l

**13.Create a listing of the years in which employees were born in, along with number of employees born in that year.**

cut -f3 newfriends| uniq |wc –l

**LAB 05. Vi Editor**

**5.1: Working wth Vi Editor**

1. **Create a file using Vi. Enter the following text: A network is a group of computers that can communicate with each other, share resources, and access remote hosts or other networks. Netware is a computer network operating system designed to connect, manage, and maintain a network and its services. Some of the network services are Netware Directory Services (NDS), file system, printing and security.**

vi editor.txt

A network is a group of computers that can communicate with each other, share resources (such as hard disks and printers), and access remote hosts or other networks. Netware computer network operating system designed to connect, manage, and maintain a network and its services. Some of the network services are Netware Directory Services (NDS), file system, printing and security.

1. **Change the word “Netware” in the second line to “Novell Netware”.**

**ESC + R**

**b. Insert the text “(such as hard disks and printers)” after “share resources” in** the first line.

ESC + i

**c. Append the following text to the file:**

“Managing NDS is a fundamental administrator role because NDS provides a single point for accessing and managing most network resources.”

ESC + A

**2: Create the data files, used in the previous lab sessions using vi editor.**

vi filename.txt

vi friends

vi e.txt

**LAB 6. SED Commands**

**6.1: Using SED Commands**

**1. Create a file “Employee.dat” with text as follows.**

James 76382 PACE Chennai

John 34228 GRIT Hyderabad

Peter 22321 GE Bangalore

Albert 32342 GRIT Pune

Mathew 23222 PACE Mumbai

Richard 23232 ACS Pune

**a) Write a sed command to print only the lines starting at line 2 and ending with the letters “Pune”**

**b)Write a sed command that will display the top 5 lines from the file**

**-bash-3.2$ sed "5q" Employee.dat**

James 76382 PACE Chennai

John 34228 GRIT Hyderabad

Peter 22321 GE Bangalore

Albert 32342 GRIT Pune

Mathew 23222 PACE Mumbai

**c) Write a sed command that will substitute the word “Chennai” for "Pune" used in all instance of the word**

sed "s/Pune/Chennai/" Employee.dat

James 76382 PACE Chennai

John 34228 GRIT Hyderabad

Peter 22321 GE Bangalore

Albert 32342 GRIT Chennai

Mathew 23222 PACE Mumbai

Richard 23232 ACS Chennai

**d) Write a sed command that will replace occurrence of the character e with the string UNIX in all lines. (Use –e option)**

sed "s/e/Unix/" Employee.dat

JamUnixs 76382 PACE Chennai

John 34228 GRIT HydUnixrabad

PUnixter 22321 GE Bangalore

AlbUnixrt 32342 GRIT Pune

MathUnixw 23222 PACE Mumbai

Richard 23232 ACS PunUnix

**e) Write a sed command to delete blank lines**

sed '/^$/d' Employee.dat

James 76382 PACE Chennai

John 34228 GRIT Hyderabad

Peter 22321 GE Bangalore

Albert 32342 GRIT Pune

Mathew 23222 PACE Mumbai

Richard 23232 ACS Pune

**f)Write a sed command to delete lines from 3 to 5**

sed "3,5d" Employee.dat

James 76382 PACE Chennai

John 34228 GRIT Hyderabad

Richard 23232 ACS Pune

**2: Create a new file “PACE.dat which has only the lines that contain the word** **“PACE” from Employee.dat**

sed -n "/PACE/p" Employee.dat >PACE.dat

cat PACE.dat

James 76382 PACE Chennai

Mathew 23222 PACE Mumbai

**LAB 07-Process related commands**

## 7.1: Using Process-Related Commands

1. **Find out the PID of the processes that are activated by you**

$ ps -u

1. **Find out the information about all the processes that are currently active**

$ ps -u -l

1. **Start a different process in the background. Find out the status of the background process** **using the PID of the same.**

$ sort –o emp.lst emp.lst &

$ ps -u -l

$ ps 16518

**LAB 08- AWK**

## 8.1: Writing awk-Scripts

**1. Consider the results are stored in following format:**

**EmpID Name Subject Marks(/50)**

**E001 Nilesh Unix 30**

**E002 Suresh DSA 20**

**Like these you have 10 records ( 5 of DSA and 5 of Unix)**

**1) Display First 5 records.**

$ cat > emplyee.dat

EmpId Name Subject Marks(/50)

E001 Nilesh Unix 30

E002 Suresh DSA 20

E003 Ramesh Unix 40

E004 Rahul DSA 10

E005 Sam Unix 35

E006 Jack Unix 45

E007 Arvind DSA 38

E008 Neha Unix 48

E009 Pooja DSA 20

E010 Nancy DSA 50

**2) Display Employees who belongs to subject Unix**

$ awk '$3 == "Unix"' emplyee.dat

E001 Nilesh Unix 30

E003 Ramesh Unix 40

E005 Sam Unix 35

E006 Jack Unix 45

E008 Neha Unix 48

**3) Display employee names that got less than 30 marks in DSA subject.**

$ awk '$3=="DSA" && $4&lt;30' emplyee.dat

E002 Suresh DSA 20

E004 Rahul DSA 10

E009 Pooja DSA 20

**4) Display only even numbered records**

$ awk 'NR % 2' emplyee.dat

EmpId Name Subject Marks(/50)

E002 Suresh DSA 20

E004 Rahul DSA 10

E006 Jack Unix 45

E008 Neha Unix 48

E010 Nancy DSA 50

**5) Display Employee id of records whose name start with N**

$ awk '$2~/N/' emplyee.dat

EmpId Name Subject Marks(/50)

E001 Nilesh Unix 30

E008 Neha Unix 48

E010 Nancy DSA 50

**LAB 09- SHELL SCRIPT**

**1: Writing Shell-Scripts**

**1. Display the Primary and Secondary prompt. Change the primary prompt to your name: temporarily**

$ echo $PS1

[\u@\h \W]\$

$ echo $PS2

>

PS1="Yash#"

Yash#

**2: As soon as you login, the prompt should be changed to your name: also the name of the home directory should be automatically displayed.**

PS1="Yash#"

Yash#

**3: Check the content of the Environmental variable SHELL.**

Yash#echo "$SHELL"

/bin/bash

**4: Try the below exercise and check the output.**

Note: Type every line and press enter, do not type the entire code in a vi editor.

$continent=”Africa”

$echo “$continent”

------------? Africa

$sh

$echo “$continent”

------------? No Response

$continent=”Asia”

$echo “$continent”

------------? Asia

$ctrl + d

$echo “$continent”

------------? Africa

$sh

$echo “$continent”

------------? No Response

$ctrl + d

**5: Try the below exercise and check the output. (Export variables)**

Note: Type every line and press enter, do not type the entire code in a vi editor.

$continent=”Africa”

export continent

$echo “$continent”

------------? Africa

$sh

$echo “$continent”

------------? Africa

$continent=”Asia”

$echo “$continent”

------------? Asia

$ctrl + d

$echo “$continent”

------------? Africa

**6: Write a shell script that takes the user name as input and reports whether he / she has logged in or not.**

r=$(who|grep "$1")

if [ -n "$r" ]

then

echo "$1 logged in"

echo $r

else

echo "$1 isn't logged in"

fi

$ sh userlog.sh ssis16

root logged in

root :0 2017-07-11 23:44 (:0) ssis16 pts/0 2017-07-12 00:54 (:0)

sh userlog.sh Hari

Hari isn't logged in

**7: Write a shell script to display the file name and its contents of all the files that is there in the current directory.**

$ more \* | cat

**8: Write a shell script, which will take a file name as argument and check whether the file exists and display its access permissions for user.**

file.sh

if [ -e "$1" ]

then

echo "$1 File Found"

else

echo "$1 File not found"

fi

**9: Pass three numbers as command line arguments and display the largest number in the given three numbers.**

if [ $1 -gt $2 ]; then

if [ $1 -gt $3 ]; then

echo "$1 is the largest"

else

echo "$3 is the largest"

fi

elif [ $2 -gt $3 ]; then

echo "$2 is the largest"

else

echo "$3 is the largest"

fi

**10: Write a shell script which will accept a pattern and a file name. The pattern will be searched in the file provided. Display appropriate messages and perform necessary validations on file.**

echo "Enter pattern: "

read pat

echo "Enter filename: "

read filename

if [ -f $filename ]

then

op=$(grep $pat $filename);

if [ -n "$op" ]

then

echo "$op"

else

echo "Pattern not found."

fi

else

nano patfile.sh

sh patfile.sh

Enter pattern:

Char

Enter filename:

friends.txt

**7.Charu Bhandari Goregaon**

$ sh patfile.sh

Enter pattern:

CCC

Enter filename:

friends.txt

Pattern not found.

nano patfile.sh

sh patfile.sh

Enter pattern:

Xhar

Enter filename:

gds

File does not exist

**11: To create a menu program for a) creating a file, b) Creating a directory, c) copying a file, d) moving a file. (use functions)**

a. If the file exists already give the appropriate message

b. If the dir exists already give the appropriate error message

c. Source file should exist if not give a message, It should have read permission if not another message, Destination file either there or not,

if not there then create it and copy it.

If there, then ask whether to overwrite or not,

if yes then overwrite it or else give a message file exists already and not overwritten.

create\_file&#40;&#41;{

echo "Enter file name"

read file

if [ ! -f $file ]; then

touch $file

echo "Successfully created file $file"

else

echo "File already exists! Do you want to override the file?(Y/N)"

read ans

if [ $ans -eq "Y" -o $ans -eq "y" ]; then

touch $file

echo "Successfully created file $file"

else

echo "Not creating file $file...";

fi

fi

}

create\_dir(){

echo "Enter directory name"

read dir

if [ ! -d $dir ]; then

mkdir $dir

else

echo "Directory already exists!"

fi

}

cp\_file&#40;&#41;{

echo "Enter source file name"

read source

echo "Enter destination file name"

read des

if [ -f $des ]; then

echo "Files exists at destination! Want to override(Y/N)"

read ans

if [ $ans -eq "Y" -o $ans -eq "y" ]; then

cp $source $des

echo "Successfully copied $source to $des"

else

echo "Not copying file...";

fi

else

cp $source $des

echo "Successfully copied $source to $des"

fi

}

mv\_file&#40;&#41;{

echo "Enter source file name"

read source

echo "Enter destination file name"

read des

if [ -f $source ]; then

if [ -f $des ]; then

echo "Files exists at destination! Want to override(Y/N)"

read ans

if [ $ans -eq "Y" -o $ans -eq "y" ]; then

mv $source $des

echo "Successfully moved $source to $des"

else

echo "Not moving file...";

fi

else

mv $source $des

echo "Successfully moved $source to $des"

fi

else

echo "Source file does not exists"

fi

}

echo "Menu"

echo "1. Create a file"

echo "2. Create a directory"

echo "3. Copy file"

echo "4. Move file"

echo "5. Exit"

read choice

case $choice in

1)create\_file;;

2)create\_dir;;

3)cp\_file;;

4)mv\_file;;

5)exit;;

esac

**12: Write a function yesno() to display question to user and accept answer as y/n. If answer to the question is y the function should return 0 otherwise 1.**

**Use yesno functions for asking different questions. Question will be passed as parameter to the function.**

**Accept filename from user check whether it is file or directory. Use yesno() function to display question do you really want to delete file? If the ans is y, then delete the file or directory.**

yesno(){

echo "Do you really want to delete this file? [y/n]"

read choice

if [ $choice != 'y' ]

then

return 0

else

rm $1

return 1

fi

}

echo "Enter filename to delete: "

read filename

yesno $filename

ret=$?

if [ $ret -eq 1 ]

then

echo "$filename deleted"

else

echo "$filename was not deleted"

fi

$ touch hello.ycyc

$ sh yesno.sh

Enter filename to delete:

hello.ycyc

Do you really want to delete this file? [y/n]

n

hello.ycyc was not deleted

$ sh yesno.sh

Enter filename to delete:

hello.ycyc

Do you really want to delete this file? [y/n]

y

hello.ycyc deleted

**13: Write a shell script to store names of four employees and check whether those employees are currently logged in or not. Display appropriate message.**

r1=$(who|grep "$1")

if [ -n "$r1" ]

then

echo "$1 logged in"

echo $r1

else

echo "$1 isn't logged in"

fi

r2=$(who|grep "$2")

if [ -n "$r2" ]

then

echo "$2 logged in"

echo $r2

else

echo "$2 isn't logged in"

fi

r3=$(who|grep "$3")

if [ -n "$r3" ]

then

echo "$3 logged in"

echo $r3

else

echo "$3 isn't logged in"

fi

r4=$(who|grep "$4")

if [ -n "$r4" ]

then

echo "$4 logged in"

echo $r4

else

echo "$4 isn't logged in"

fi

sh userlog.sh raunak rtw rtw

Hari isn't logged in

root logged in

root :0 2017-07-13 23:44 (:0) root pts/0 2017-07-12 00:54 (:0)

root logged in

root :0 2017-07-13 23:44 (:0) root pts/0 2017-07-12 00:54 (:0)

Hari isn't logged in

**14: Accept the users first and last name and the echo the entire name along with some suitable comment.**

#fl.sh

echo "Enter first name"

read fname

echo "Enter last name"

read lname

echo "Your name is$fname $lname"

**15: List all files that have been modified today.**

$ find . -mtime -1 -print

**16: Display long listing of only the regular files in the current directory.**

$ ls -p -l | grep -v /

**17: Display details of all files in the 2 “paths” accepted from user. The display should be screen by screen.**

echo "Enter First Path"

read path1

echo "Enter Second Path"

read path2

echo "-----------Path 1 Files-----------------------"

ls $path1

echo "-----------Path 2 Files-----------------------"

ls $path2

**18: Let the script display its name and its PID.**

echo "The script name is $0"

echo "The script pid is $$"

**19: Get the concatenated o/p of 2 files into a third file: Take 3 command line arguments: The first argument is the name of a destination file,**

**and the other two arguments are names of files whose contents are to be placed in the destination file.**

$ cat "$2 $3 > $1"

**20: Write a menu driven shell program to:**

**a. Display calendar of current month**

**b. Search for a pattern in all the files/subdirectories from current directory.**

**c. Count the no. of directories / sub directories in current directory**

ch='y'

while [ $ch == "y" ]

do

echo "1. Calendar of the current month"

echo "2. Search for a pattern"

echo "3. Count no. of directories/ sub-directories in directory"

echo "Enter 1, 2 or 3:"

read choice

case $choice in

1) echo "Calendar for this month"

cal;;

2) echo "Enter pattern:"

read pat

grep $pat \* -r;;

3) echo "Here is count of all the directories / sub-directories:"

ls -r \* | wc -l;;

\*) echo "Enter valid choic.";;

esac

$ sh menu.sh

1. Calendar of the current month

2. Search for a pattern

3. Count no. of directories/ sub-directories in directory

Enter 1, 2 or 3:

1

Calendar for this month

July 2017

Su Mo Tu We Th Fr Sa

1

2 3 4 5 6 7 8

9 10 11 12 13 14 15

16 17 18 19 20 21 22

23 24 25 26 27 28 29

30 31

Do you want to continue?[y/n]

y

1. Calendar of the current month

2. Search for a pattern

3. Count no. of directories/ sub-directories in directory

Enter 1, 2 or 3:

2

Enter pattern:

Virag

friends.txt:2. Virag Shah Mulund

myfiles/friends.txt:2. Virag Shah Mulund

myfiles/myfriends.txt:1. Virag Shah Mulund

myfriends.txt:2. Virag Shah Mulund

result:2. Virag Shah Mulund

result:2. Virag Shah Mulund

Do you want to continue?[y/n]

y

1. Calendar of the current month

2. Search for a pattern

3. Count no. of directories/ sub-directories in directory

Enter 1, 2 or 3:

3

Here is count of all the directories / sub-directories:

60

Do you want to continue?[y/n]

n

**21: Display day of week for a given date. (ddmmyyyy)**

**If day is Monday, display message “Monday Blues”**

**Friday display message “yeh! It’s week end.”**

**Similarly display different messages for each day of the week.**

$ echo "Enter date: (dd-mm-yyyy)"

read date

DAY=$(cut -d"-" -f1<<<"$date")

MONTH=$(cut -d"-" -f2<<<"$date")

YEAR=$(cut -d"-" -f3<<<"$date")

newdate="$MONTH/$DAY/$YEAR"

day=$(date -d $newdate +'%A')

#echo $day

case $day in

"Monday") echo "Monday Blues" ;;

"Tuesday") echo "Still tuesday" ;;

"Wednesday") echo "Boring wednesday" ;;

"Thursday") echo "Nearing weekend" ;;

"Friday") echo "It's the weekend!" ;;

"Saturday") echo "One more day off!" ;;

"Sunday") echo "Monday tomorrow" ;;

\*) echo "Bye" ;;

esac

$ nano daydate.sh

sh daydate.sh

Enter date: (dd-mm-yyyy)

18-12-1995

Monday Blues

**22: Display the contents of all .lst files in the current directory.**

$ cat > abc.lst

hello

lst

cat abc.lst

hello

lst

$ cat > def.lst

another

lst

file

$ cat def.lst

another

lst

file

$ cat \*.lst

hello

lst

fileanother

lst

file

**23: Design a simple calculator, which will add/subtract/multiply/divide 2 numbers.**

**eg. $ cal 10 20 + will give o/p as 30.**

if [ "$3" == "+" ]

then

echo $(($2+$1))

elif [ "$3" == "-" ]

then

echo $(($1-$2))

elif [ "$3" == "\*" ]

then

echo $(($1\*$2))

elif [ "$3" == "/" ]

then

echo $(($1/$2))

else

echo "Enter proper input."

fi

$ sh cal.sh 3 4 -

-1

$ sh cal.sh 3 4 /

0

$ sh cal.sh 3 4 \*

Enter proper input.

$ sh cal.sh 3 4 "\*"

12

$ sh cal.sh 3 4 +

7

**24: For a student file with the following fields, rollno, name, marks, Generate 2 files ‘Pass’ and ‘Fail’ containing records of student who have passed or failed. Also count the number of students who have passed or failed.**

$ cat student

1 Yash 74

2 Nirbhay 87

3 Shubham 77

4 Shyam 78

5 Babu 43

6 Raj 49

7 Baburaj 46

$ cat marks.sh

file="student"

while IFS= read -r line

do

mark=$(echo "$line" | awk '{print $3}')

if [ $mark -gt 60 ]

then

echo "$line PASS"| awk '{print $1, $2, $3, $4}' >> pass.txt

else

echo "$line FAIL"| awk '{print $1, $2, $3, $4}' >> fail.txt

fi

done < $file

echo "PASS: "$(wc -l &lt; pass.txt)

echo "FAIL: "$(wc -l &lt; fail.txt)

$ sh marks.sh

PASS: 4

FAIL: 3

$ cat pass.txt

1 Yash 74 PASS

2 Nirbhay 87 PASS

3 Shubham 77 PASS

4 Shyam 78 PASS

$ cat fail.txt

5 Babu 43 FAIL

6 Raj 49 FAIL

7 Baburaj 46 FAIL

**25: Accept a date string from terminal and display employees born after the input date.**

file="emp.lst"

echo "Enter date [mm/dd/yyyy]:"

read newdate

day=$(date -d $newdate +%Y%m%d)

while IFS= read -r line

do

line2=$( echo "$line" | awk '{split($0,a,"|");print a[5]}' )

#echo "$line2"

todate=$(date -d $line2 +%Y%m%d)

#echo $todate

if [ $day -gt $todate ]

then

echo $line

fi

done < $file

$ sh empdate.sh

Enter date [mm/dd/yyyy]:

08/06/1981

7369|SMITH|CLERK|20|12/17/1980|800

7499|ALLEN|SALESMAN|30|02/20/1981|1600

7521|WARD|SALESMAN|30|02/22/1981|1250

7566|JONES|MANAGER|20|02/04/1981|2975

7698|BLAKE|MANAGER|30|01/05/1981|2850