LINUX SHELL SCRIPTING LABORATORY MANUAL

B.TECH (III SEM), AY: 2021-2022

Batch 2020-2024

SRI VASAVI ENGINEERING COLLEGE (Autonomous)

Pedatadepalli, TADEPALLIGUDEM-534 101, W.G. Dist.

Department of Computer Science & Engineering (Accredited by NBA)

SRI VASAVI ENGINEERING COLLEGE (Autonomous)

PEDATADEPALLI, TADEPALLIGUDEM.

Certificate

This is to certify that this is a bonafide record of Practical Work done in ***LINUX SHELL SCRIPTING Lab*** *by Mr. /Miss*_____ *bearing RollNo.*_____ *of* ***CSE*** *Branch of* ***III Semester*** *during the academic year* ***2021-22***.

No. of Experiments Done:

Faculty In charge of the Laboratory Head of the Department

EXTERNAL EXAMINER

LIST OF EXPERIMENTS:

1. Experiment the following Unix Commands:
    a) General Purpose Utilities: cal, date,man,who.
    b) Directory Handling Commands: pwd,cd,mkdir,rmdir.
    c) File Handling Utilities: cat,cp,ls,rm,nl,wc
    d) Displaying Commands: head, tail
    e) Filters: cmp,comm.,diff,sort,uniq
    f) Disk Utilities: du,df
2. Develop a Shell Program to Display all the words which are entered as command line arguments.
3. Develop a shell script that Changes Permissions of files in PWD as rwx for users.
4. Develop a shell script to print the list of all sub directories in the current directory.
5. Develop a Shell Program which receives any year from the keyboard and determine whether the year is leap year or not. If no argument is supplied the current year should be assumed.
6. Develop a shell script which takes two file names as arguments-If their contents are same then delete the second file.
7. Develop a shell script to print the given number in the reversed order.
8. Develop a shell script to print first 25 Fibbonacci numbers.
9. Develop a shell script to print the Prime numbers between the specified range.
10. Develop a shell script to delete all lines containing the word ‘unix’ in the files supplied as arguments.
11. Develop a shell script Menu driven program which has the following options.
    i) contents of /etc/passwd
    ii) list of users who have currently logged in.
    iii) present working directory.
    iv) exit.

Computer :

It is an electronic device used for storing and processing data in binary form.

Program :

Set of instructions used by computer to perform specific task.

Software :

It is a program and other operating information used by computer.

- System software: Used by OS .
- Application software: Used by users.

Operating System (OS):

OS is system software that manages computer hardware, software resources, and provides common services for computer programs.

or

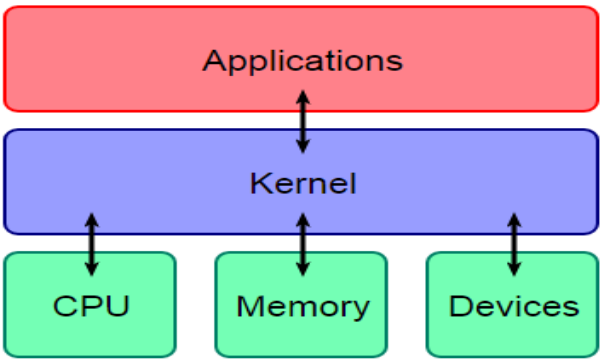
OS act as an interface between user and computer hardware.

Kernel :

It is the main core component in OS and it provides the most basic level of control over all of the computer's hardware devices. It manages memory access for programs in the RAM, it determines which programs get access to which hardware resources, it sets up or resets the CPU's operating states for optimal operation at all times, and it organizes the data for long-term non-volatile storage with file systems on such media as disks, tapes, flash memory, etc.

It has 5 services

- Memory management
- Device managament
- Process management
- Interrupt handling
- Input/Output communication



User Interface or Shell :

Every computer that is to be operated by an individual requires a user interface. The user interface is usually referred to as a shell and is essential if human interaction is to be supported.

The user interface views the directory structure and requests services from the operating system that will acquire data from input hardware devices, such as a keyboard, mouse and requests operating system services to display prompts, status messages and such on output hardware devices, such as a video monitor or printer.

The two most common forms of a user interfaces are:

- Command Line Interface(CLI): where computer commands are typed out line-by-line.
- Graphical User Interface(GUI): where a visual environment is present. A mouse is used to navigate the computer.

UNIX :

UNICS(UNiplexed Information Computing System) is developed in 1969 by Ken Thompson at AT&T Bell labs,firstly developed in assembly language and then developed in C language by Dennis Ritchie.

Features of Unix OS :

- Unix is a portable,multi-user, multitasking operating system.
- It can be used as the master control program in workstations and servers.
- Hundreds of commercial applications are available.

Advantages :

- Portable,supports hierarhical file system
- Secure due to its strong server validation and authentication.

Disadvantages :

- CLI usage difficulty.
- Documentation of various Unix tools is hard to find.

Examples :

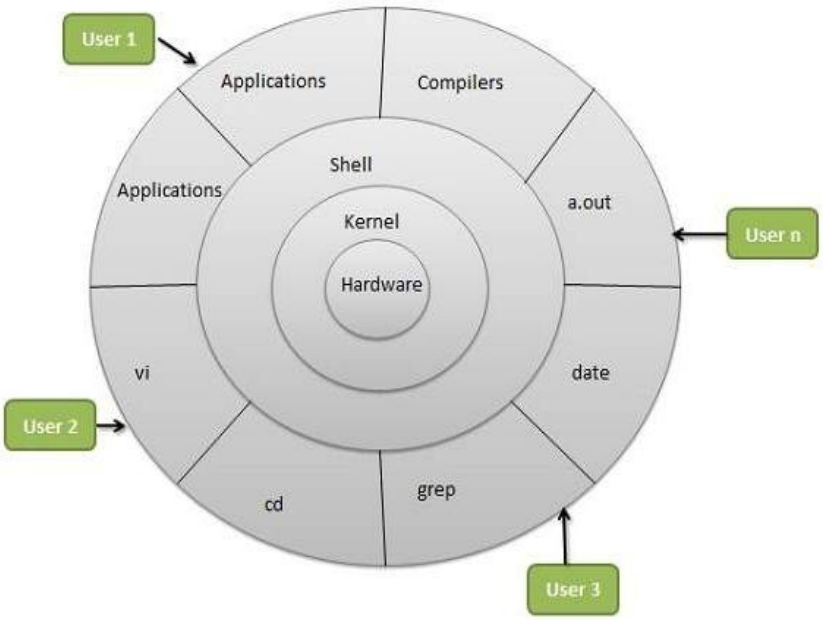
- Solaris
- AIX
- FreeBSD
- NetBSD
- Xenix

- Macos

LINUX :

Loveable Intellect Not Using XP.It is developed by Linus Torvalds in 1991 at Linux community.It is Unix like OS developed by using C language.

Architecture of Linux :



Features of Linux OS :

- Portable,support multitasking.
- Programs consist of one or more processes, and each process have one or more threads
- It can easily co-exists along with other Operating systems.
- Linux can run multiple user programs.
- Individual accounts are protected because of appropriate authorization.

Advantages :

- Compatible with large number of file format.
- Open source software and is free of cost.
- User can make changes in Linux OS.
- Ensures privacy of user as it doesnot collect much user data.

Disadvantages :

- Use of CLI.
- Some graphic tools are not avilable.

- It doesnot have standard versions,so it is difficulty for users to choose the best version for their needs.

Examples :

- Fedora
- Ubuntu
- Slackware
- Debian.

Key differences between Unix and Linux :

- Linux source code is available to the general public whereas Unix source code is proprietary.
- Linux is a clone of Unix
- Linux default shell is BASH while the Unix shell is Bourne Shell.
- Linux threat detection and solution are very fast while Unix users require longer wait times to get the proper bug fixing patch.

Command :

A command is a directive to a computer program to perform a specific task. It may be issued via a command-line interface, such as a shell, or as input to a network service as part of a network protocol, or as an event in a graphical user interface triggered by the user selecting an option in a menu.

Shell Script :

A shell script is a computer program designed to be run by the Unix/Linux shell which could be one of the following:

- The Bourne Shell
- The C Shell
- The Korn Shell
- The GNU Bourne-Again Shell

A shell is a command-line interpreter and typical operations performed by shell scripts include file manipulation, program execution, and printing text.

Use of LSS lab :

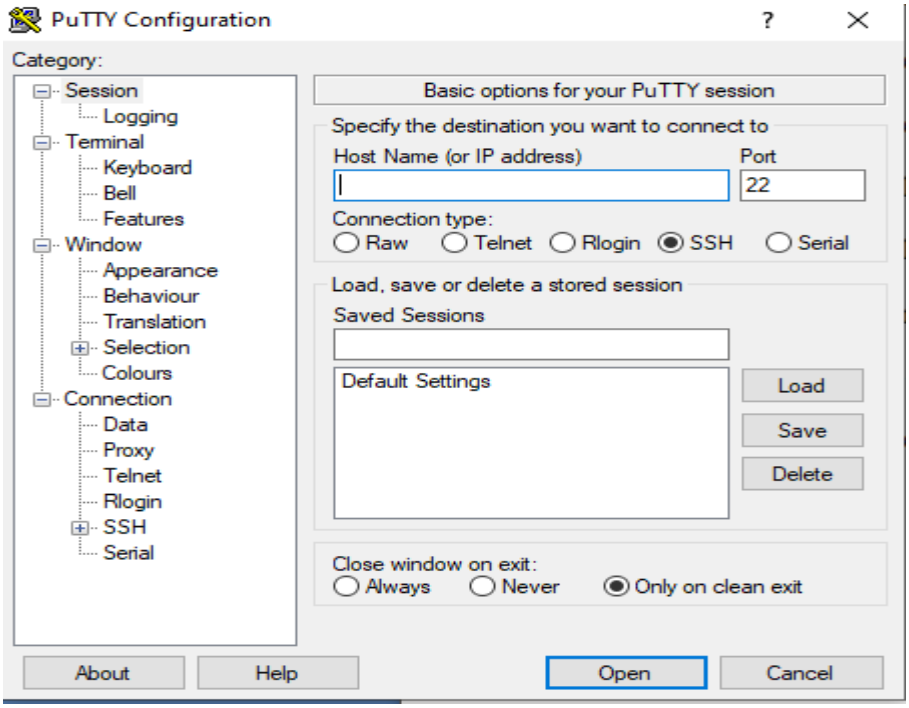
The Linux command is a utility of the Linux operating system. All basic and advanced tasks can be done by executing commands. The commands are executed on the Linux terminal. The terminal is a command-line interface to interact with the system, which is similar to the command prompt in the

Windows OS. Commands in Linux are case-sensitive.Many of the tasks that someone would like to perform on a computer are regular, require repetition, or are menial or tedious to do by hand. Shell scripting allows one to interact programmatically with a shell to do certain tasks.

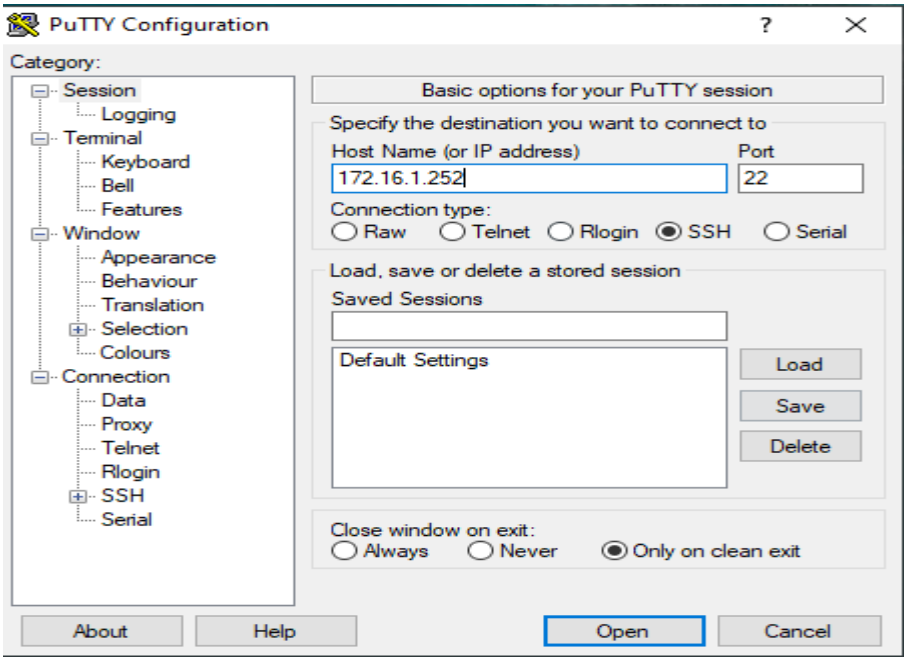
PuTTY allows the use of SSH (Secure Shell) to access a remote computer. It is a software terminal emulator that supports VT100 emulation, telnet, SSH, kerberos, and serial port connections.

Procedure to connect to LINUX:

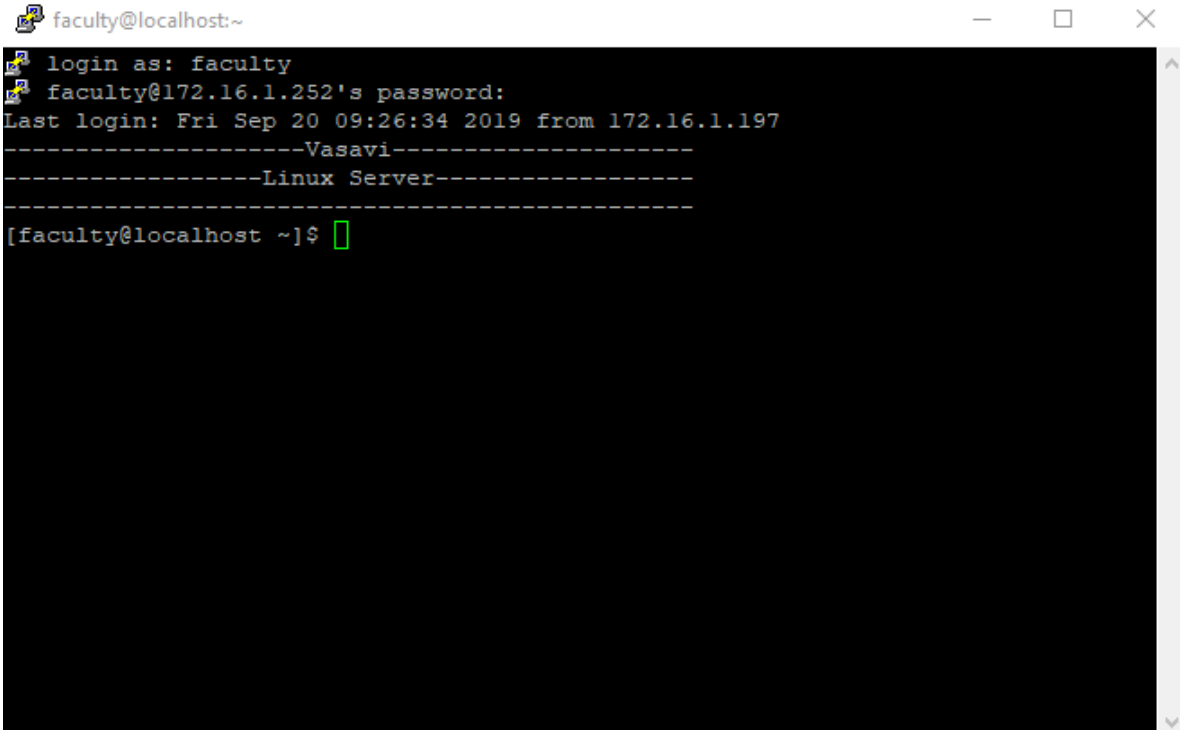
Step 1: Click on putty icon available on desk top. A window is opened



Step 2: Fill in IP address of linux server and click open.



Step 3: Provide login and password (nothing is displayed on screen while typing password) .



Step 4: Change the default password at your first login.(not for students).

Step 5: Logout from the system using either logout or exit command.

PROGRAM -1 **Date:**

Aim: Experiment the following Unix Commands:

a) General Purpose Utilities: cal, date,man,who.

b) Directory Handling Commands: pwd,cd,mkdir,rmdir.

c) File Handling Utilities: cat,cp,ls,rm,nl,wc

d) Displaying Commands: head, tail

e) Filters: cmp,comm.,diff,sort,uniq

f) Disk Utilities: du,df

a) General Purpose Utilities: cal, date,man,who.

cal(calendar) command: It displays calendar for specified month or a year.It has no options but uses arguments and the arguments are optional.

a. if no arguments are entered ,the calendar for the current month is printed

   syntax:

   $ cal

   Example:

   $ cal

b. if only one argument is entered,it is assumed to be a year and the calendar for that year is displayed.

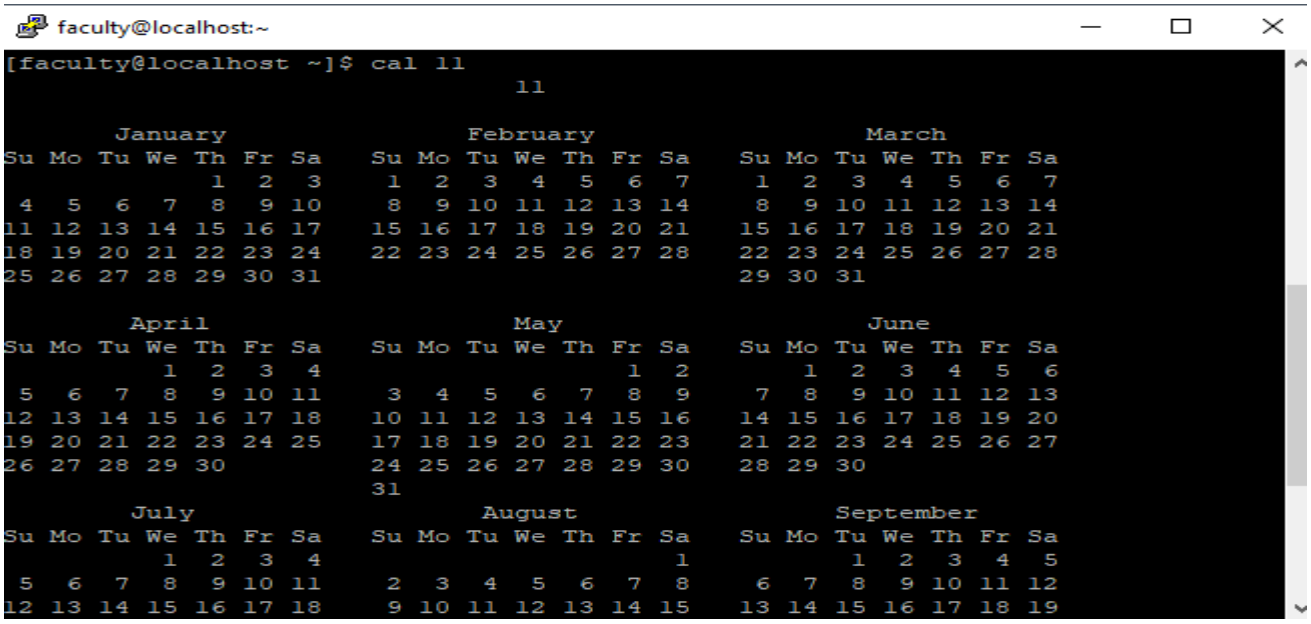
   Syntax:

   $ cal argument1

   Example:

   $ cal 11

   $ cal 2021



c. if both arguments are entered,then just one month is displayed

Syntax:

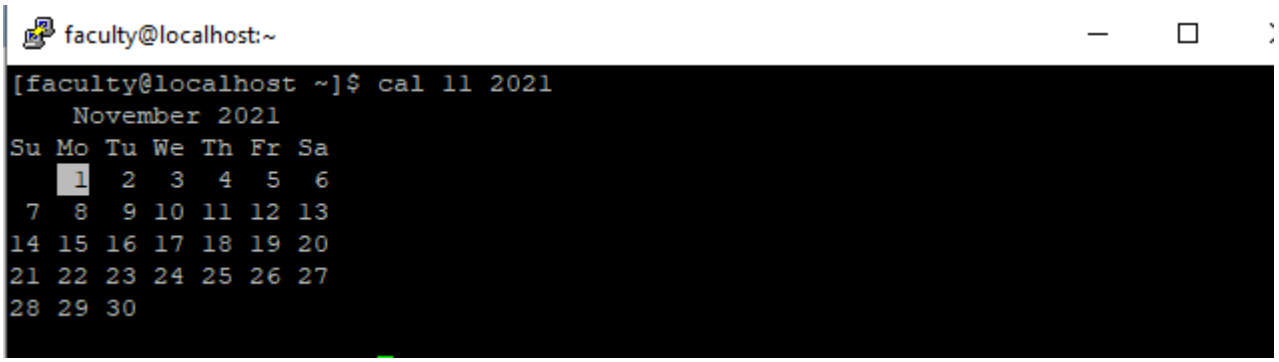
$ cal argument1 argument2

example:

$ cal 11 2021

$ cal 01 21

$ cal 11 2



date(date and time) command: It displays the system date and the time.If the system is local then it displays current system time.If the system is remote then it displays the time where the system is physically located.It can be used to set the date and time,but only by a system administrator.

Syntax:

$ date options arguments

Options:

-u universal

Arguments:

a abbreviated weekday name

A abbreviated full weekday name

b abbreviated month name

B abbreviated full month name

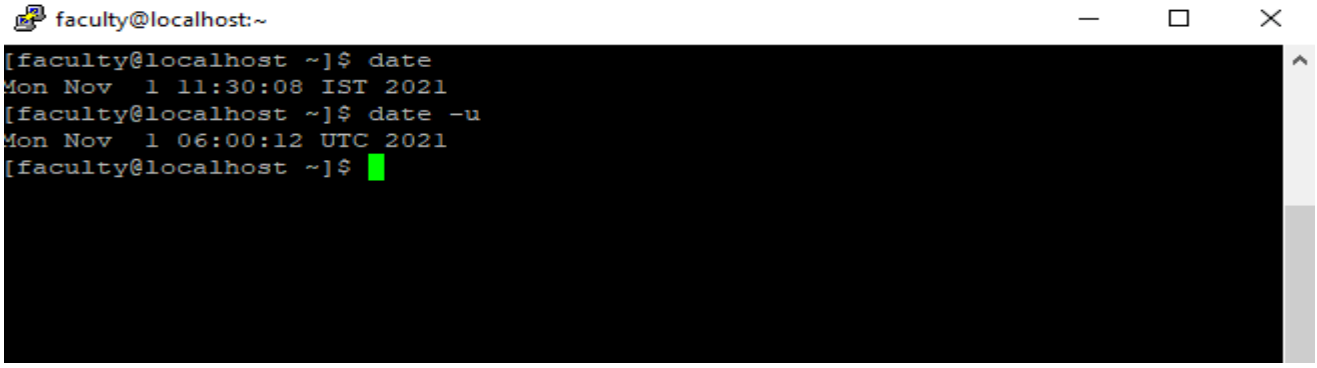
d day of the month with two digits(leading zeros)

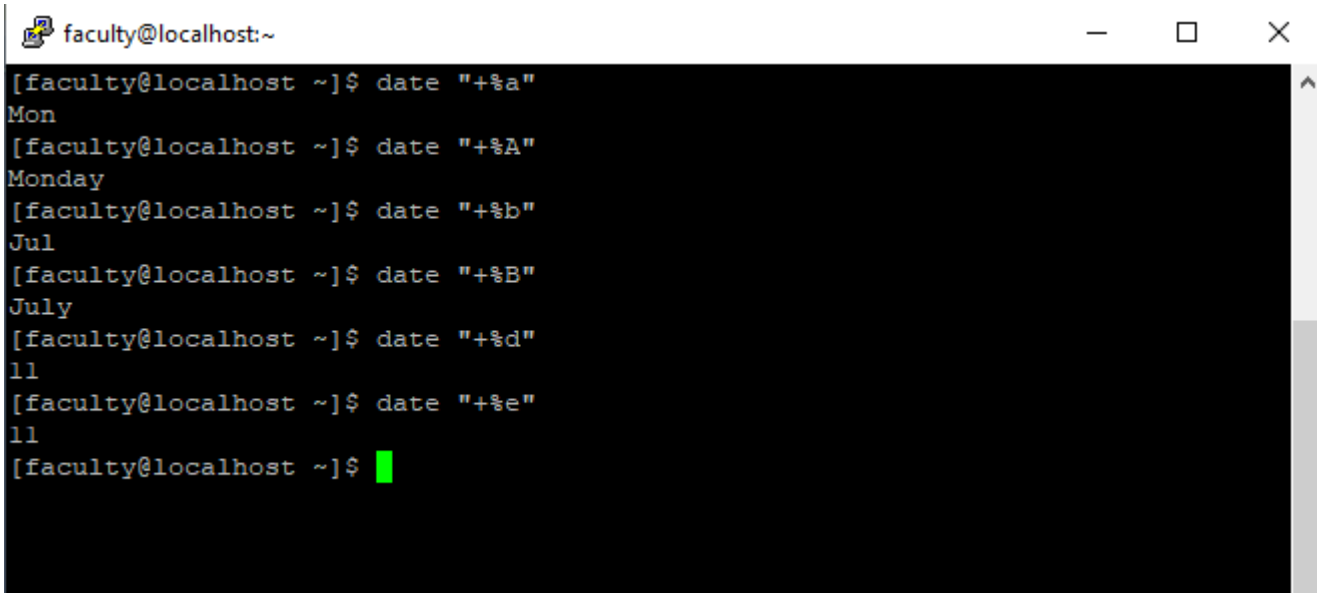
e day of the month with spaces replacing leading zeros

Example:

$ date

$ date –u





man(online documentation) command: It is one of most important command and it displays online documentation of a command when you can't remember exactly what the options are for a command.There is even a manual explanation for the man command itself.

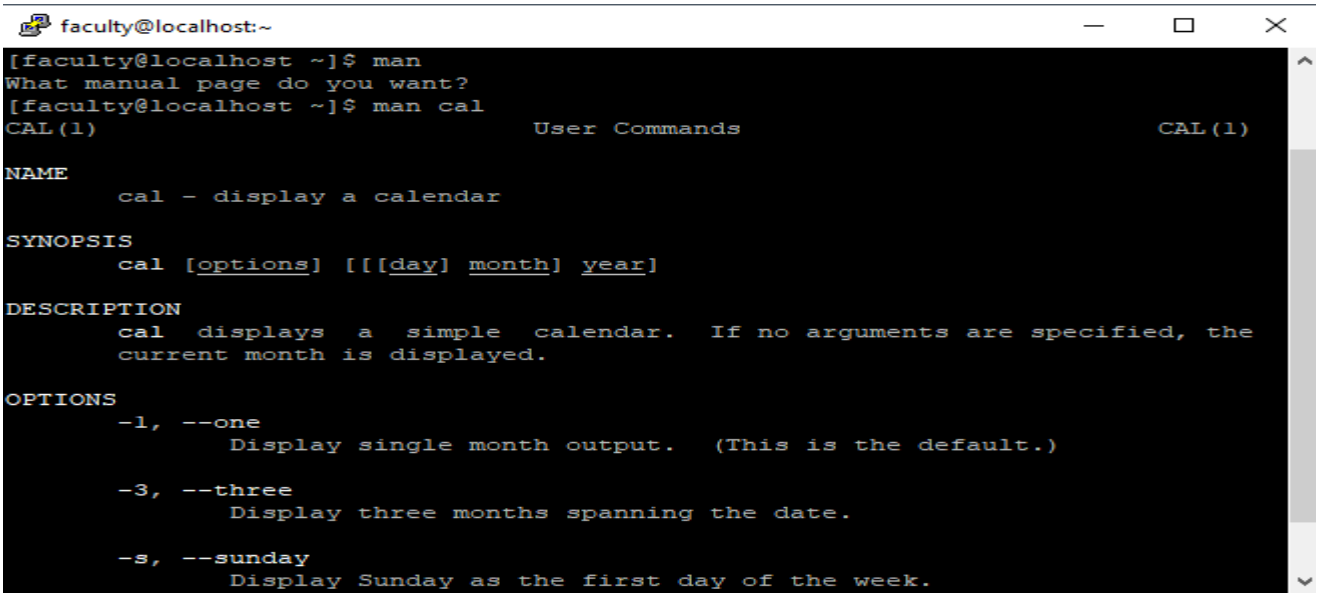
Syntax:

$ man [any command]

Example:

$ man cal

$ man man



who(who's online) command: It displays all users currently logged into the system.The who command returns the user's name(id),terminal and the time he/she logged in.

Syntax:

$ who options

Options:

-u	same as who command with idle times
-uH	same as who –u with header

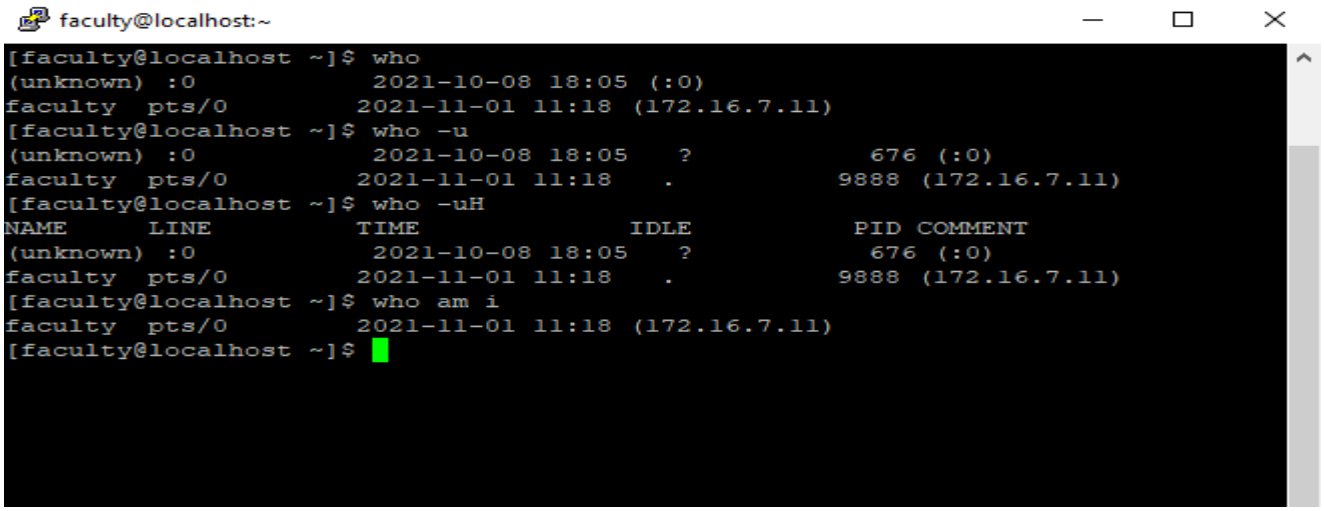
Example:

$ who

$ who -u

$ who –uH

$ who am i:It is used to ask about ourselves and it returns our user name(id).



b) Directory Handling Commands: pwd,cd,mkdir,rmdir.

pwd(locate directory / print working directory) command: It is used to determine the current directory is print working directory.It has no options and no attributes or arguments.When executed ,it prints the absolute path name for the current directory.

Syntax:

$ pwd

Example:

$ pwd

cd(change directory) command:cd command is used to change our working directory.If we have multiple directories,then we need some way to move among them.There are no options in cd command but have pathname

i.e either relative or absolute path.Generally it is relative.If there is no pathname argument,the target is the home directory.the home directory can also be targeted by using the home abbreviation(cd ~) or cd.

Syntax:

$ cd pathname

Example:

$ cd LSS

mkdir(make directory) command: It is used to make or create a new directory.

Syntax:

$ mkdir options directoryname

Options:

-m permission mode

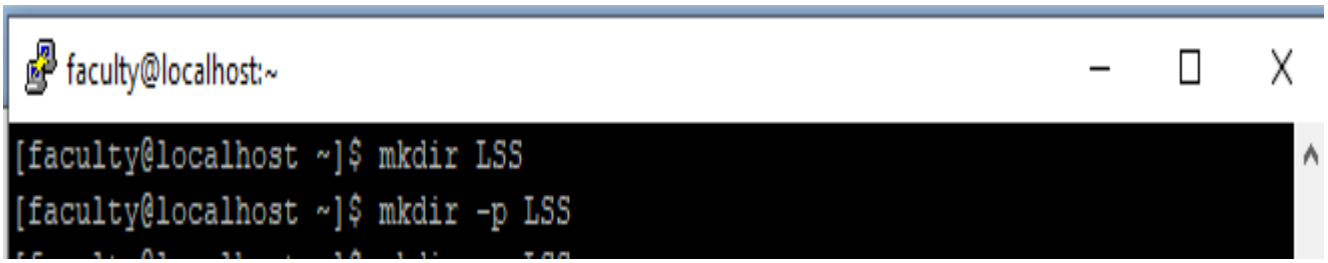
-p parent directories

Example:

$ mkdir LSS

$ mkdir –m LSS

$ mkdir –p LSS



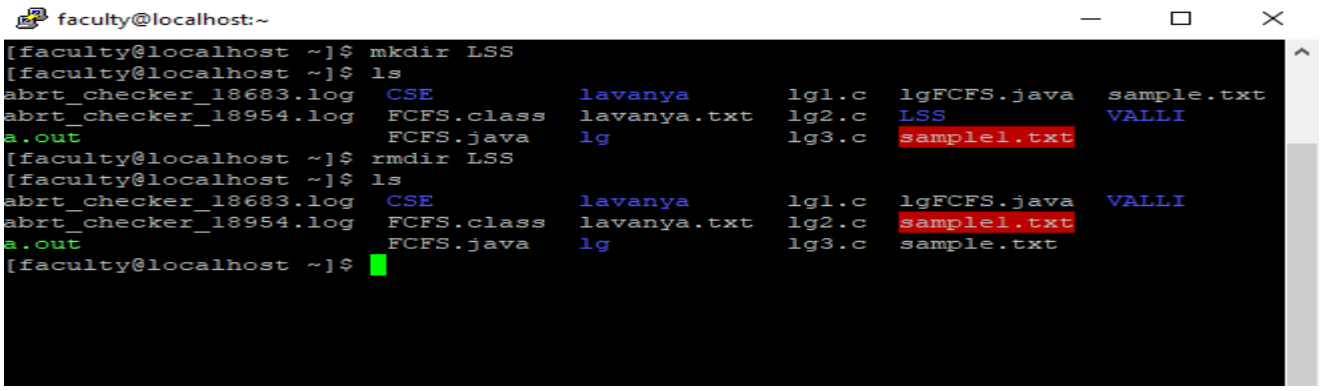
rmdir(remove directory) command: It is used to remove a directory which is no longer required.It canot delete a directory unless it is empty.If it contains any files,then “directory not empty” error message is displayed.It has no options.

Syntax:

$ rmdir directoryname

Example:

$ rmdir LSS



c) File Handling Utilities: cat,cp,ls,rm,nl,wc

cat(catenate) command: cat linux command concatenates files and print it on the standard output.

syntax: cat options inputfiles

options:

-A Show all.

-b Omits line numbers for blank space in the output.

-e A $ character will be printed at the end of each line prior to a new line.

-E Displays a $ (dollar sign) at the end of each line.

-n Line numbers for all the output lines.

-s If the output has multiple empty lines it replaces it with one empty line

-T Displays the tab characters in the output.

-v Non-printing characters (with the exception of tabs, new-lines & form-feeds) are printed visibly.

Operations With cat Command:

1. To Create a new file: $cat > file1.txt

This command creates a new file file1.txt. After typing into the file press control+d (^d) simultaneously to end the file. Make sure you type '**Ctrl-d'** at the end to save the file.

2. To Append data into the file: $cat >> file1.txt

To append data into the same file use append operator >> to write into the file, else the file will be overwritten (i.e., all of its contents will be erased). Make sure you type '**Ctrl-d'** at the end to save the file.

3. To display a file: $cat file1.txt

This command displays the data in the file.

4. To concatenate several files and display: $cat file1.txt file2.txt

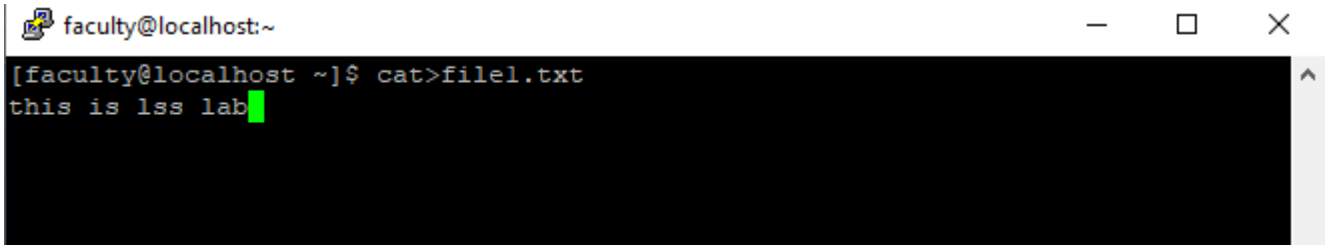
The above cat command will concatenate the two files (file1.txt and file2.txt) and it will display the output in the screen. Some times the output may not fit the monitor screen. In such situation you can print those files in a new file or display the file using less command. cat file1.txt file2.txt | less

5. To concatenate several files and to transfer the output to another file. $cat file1.txt file2.txt > file3.txt

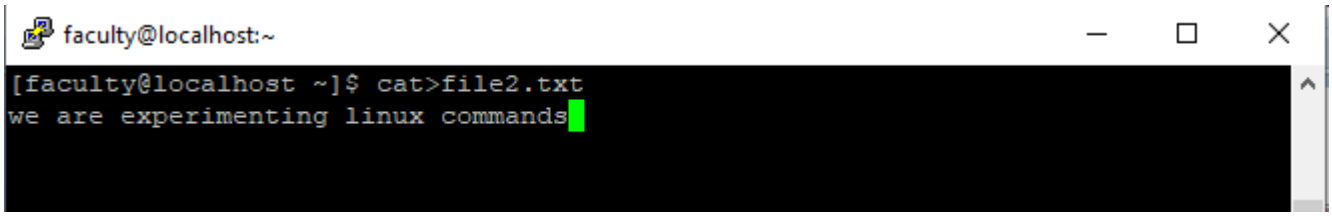
In the above example the output is redirected to new file file3.txt. The cat command will create new file file3.txt and store the concatenated output into file3.txt

Example:

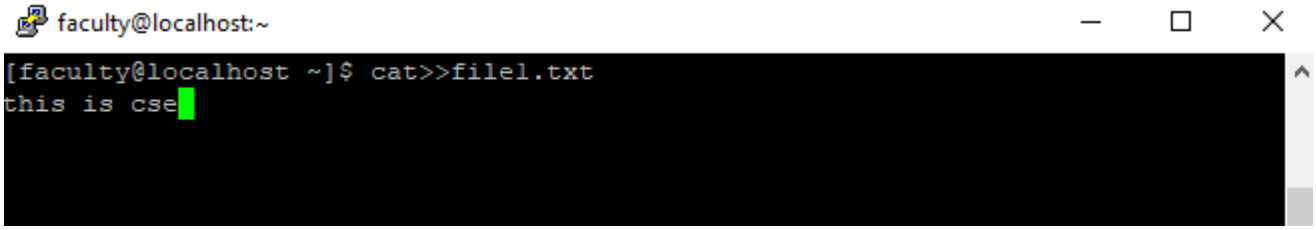
$ cat > file1.txt



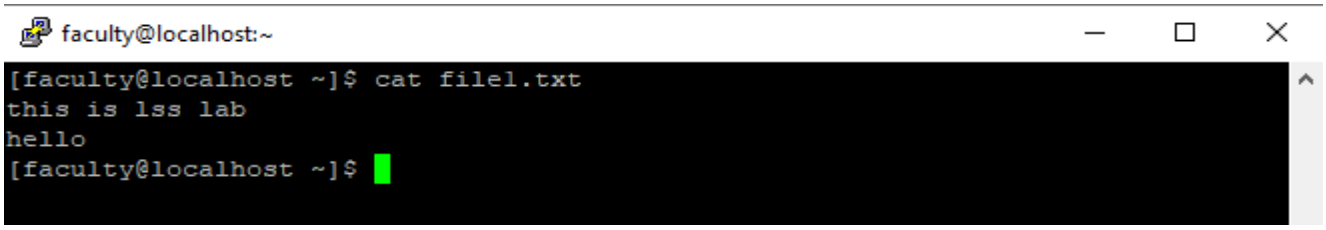
$ cat > file2.txt



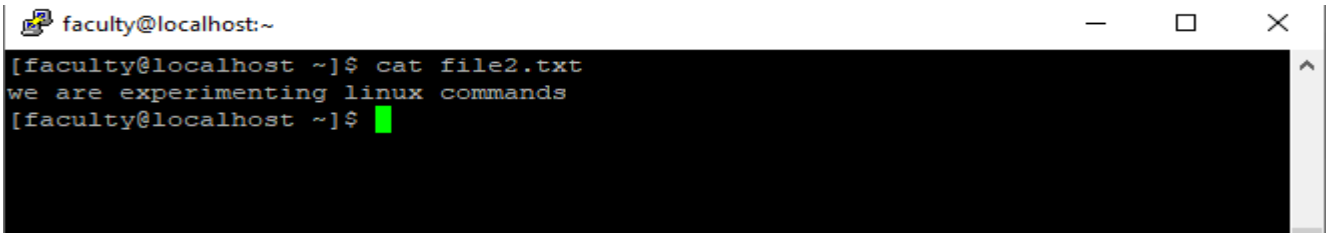
$ cat >> file1.txt



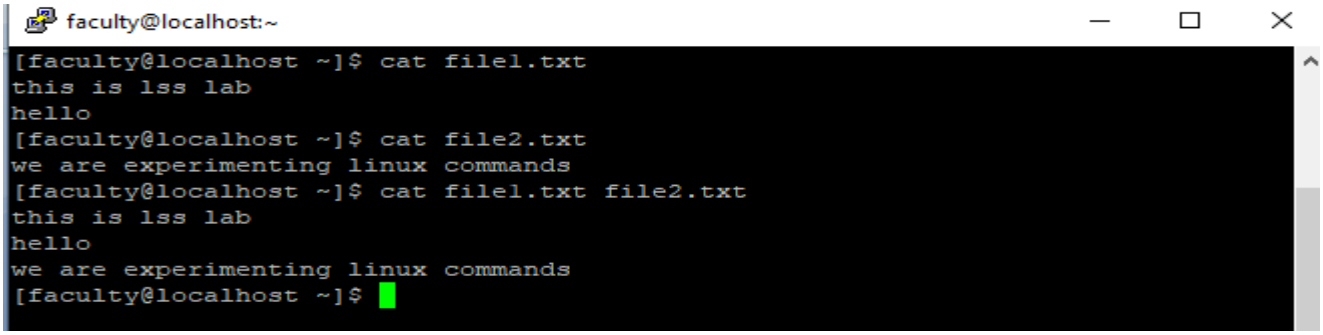
$ cat file1.txt



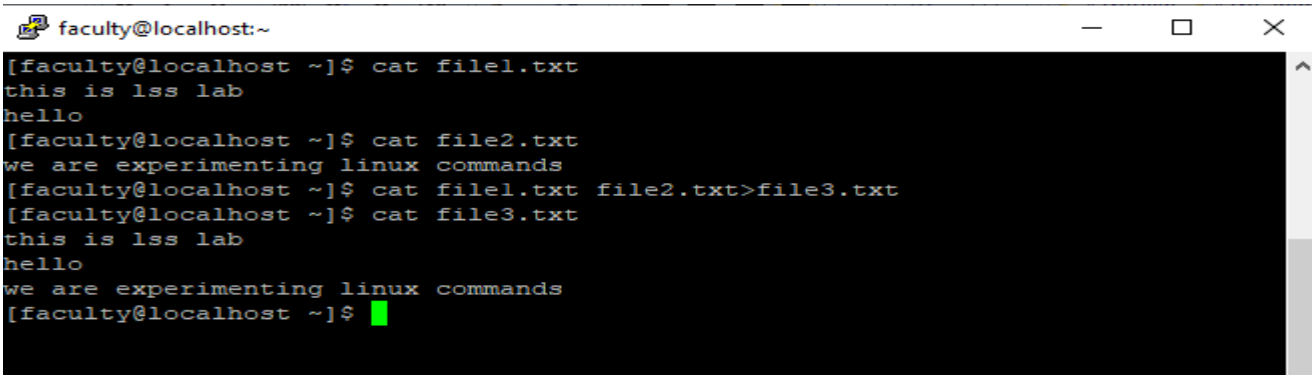
$ cat file2.txt



$ cat file1.txt file2.txt



$ cat file1.txt file2.txt>file3.txt



cp(copy) command: It creates a duplicate of a file ,a set of files or a directory.The cp command copies both text and binary files.If the source is a file,the new file contains an exact copy of the data in the source file.If the source is a directory,all of the files in the directory are copied to the destination,which must be a directory.If the destination file exists,its contebts are replaced by the source file contents.

Syntax:

$ cp options source destination

Options:

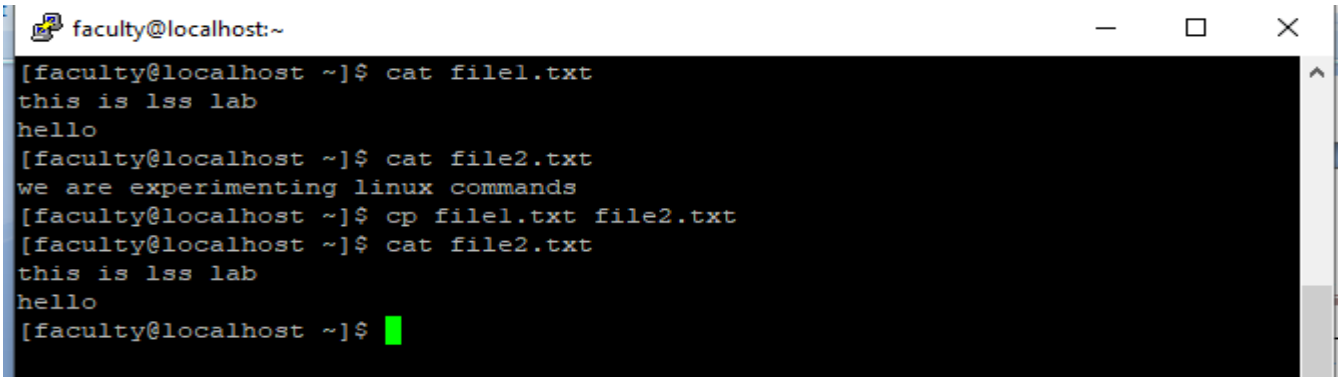
-p preserve time and permissions

-i interactive prompt

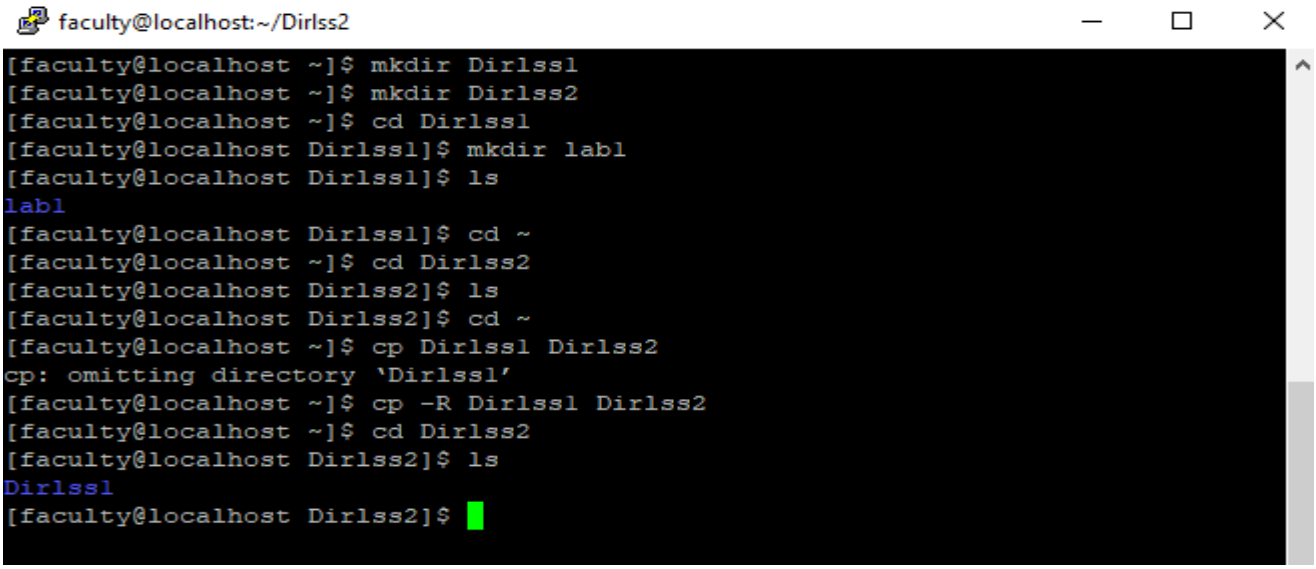
-r recursive copy subdirectories

Example:

$ cp file1.txt file2.txt



$ cp Dirlss1 Dirlss2



$ cp –p file.txt file2.txt

$ cp –i file1.txt file2.txt

ls(list) command:ls command is used to list the files or contents present in a directory.Depending on the options it can list files,directories,subdirectories.If the name of the directory is provided then it displays the contents of the working directory.

Syntax:

$ ls

Options:

-l longlist

-d working directory

-n user/group id's

-r reverse order

-t time sequence

-p identify directories

-1(0ne) print one column

-i print inode

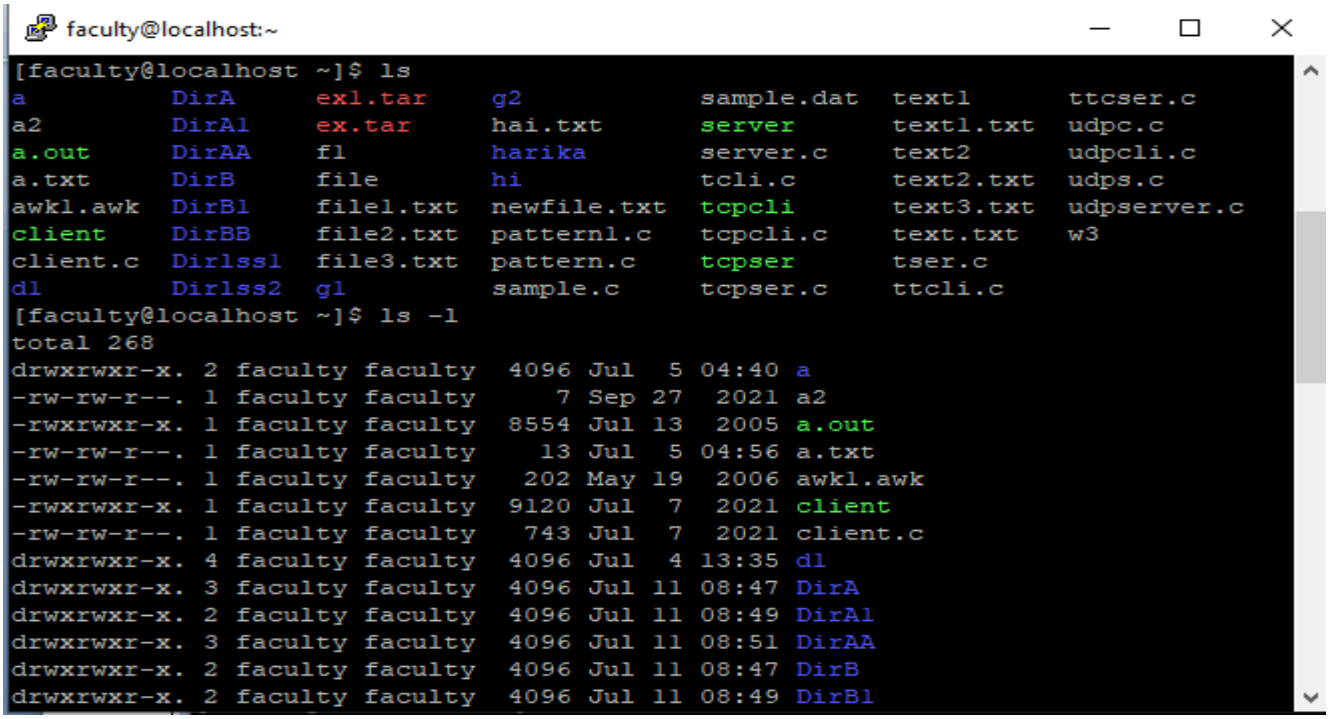
-c time inode date

-R recursive

Example:

$ ls

$ ls –l



rm(remove) command: It deletes an entry from a directory by destroying its link to the file.Remember,however,that there can be multiple links to a physical file.This means that a remove does not always physically delete a file.The file is deleted only if,after remove,there are no more links to it.To delete a file,we must have write permissions.

Syntax:

$ rm options filelist

Options:

-f force

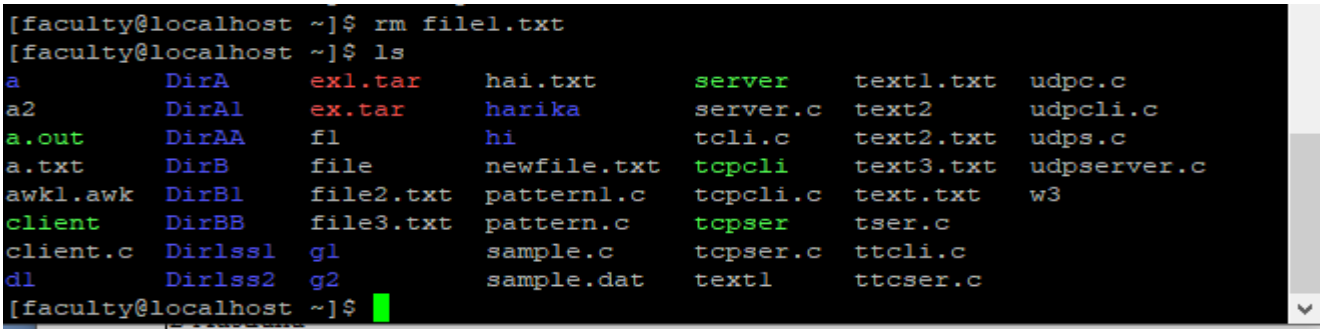
-i interactive

-r recursive

Example:

$ rm file1

$ rm –r file2

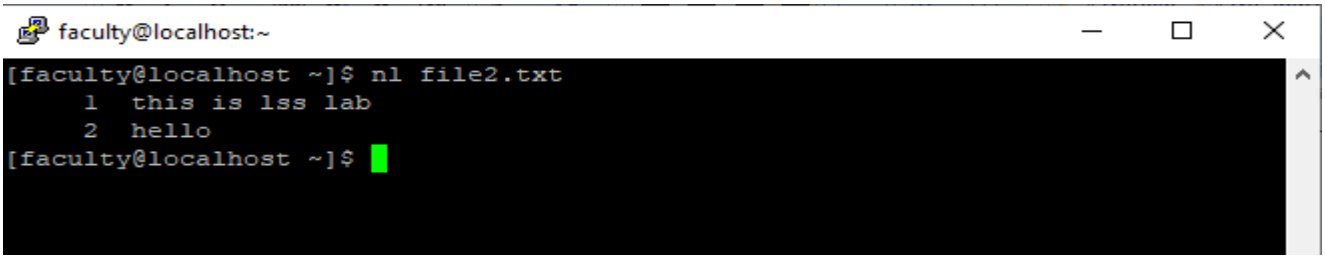


nl(number line) command: The nl utility in Linux is used to give number lines of a file on console.

Syntax:

$ nl filename

Example: $ nl file2.txt



wc(word count) command: It counts the number of characters,words and lines in one or more documents.The character count includes newlines(/n).Options can be used to limit the output to one or two of the counts.

Syntax:

wc options inputfiles

options:

-c character count

-l line count

-w word count

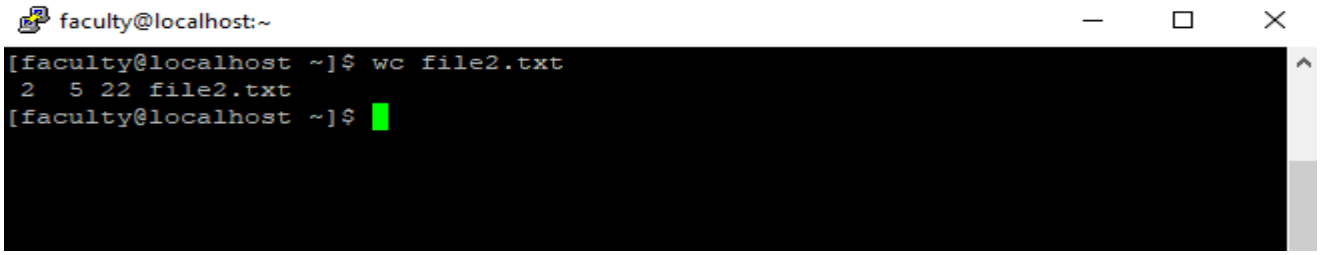
Example:

$ wc file2.txt

$ wc –c file2.txt

$ wc –l file2.txt

$ wc –w file2.txt



d) Displaying Commands: head, tail

head command: It copies a specified number of lines from the beginning of one or more files to the standard ouput stream or to displays portions of files.If no files are specified,it gets the lines from standard input.

Syntax:

head options inputfiles

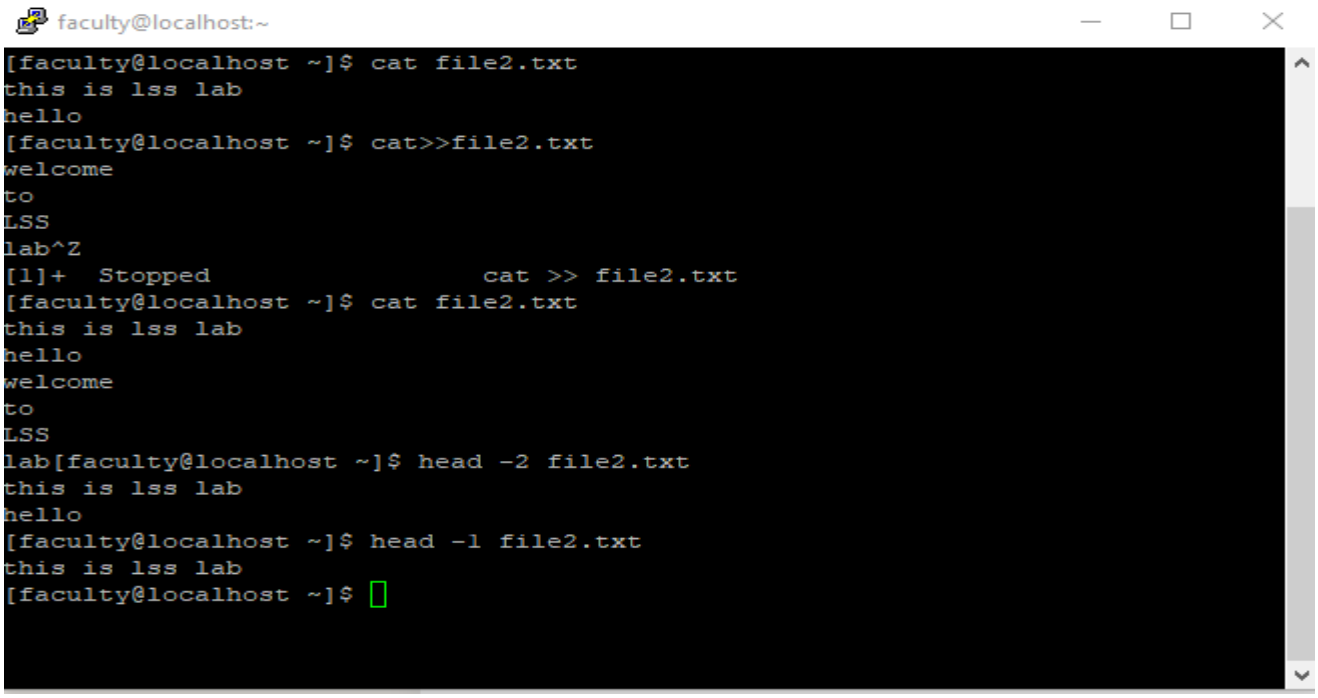
options:

-N number of lines

Example:

$ head -2 file2.txt

$ head -1 file2.txt



tail command: It copies a specified number of lines from the ending of one or more files to the standard ouput stream or to displays portions of files.If no files are specified,it gets the lines from standard input.

Syntax:

tail options inputfiles

options:

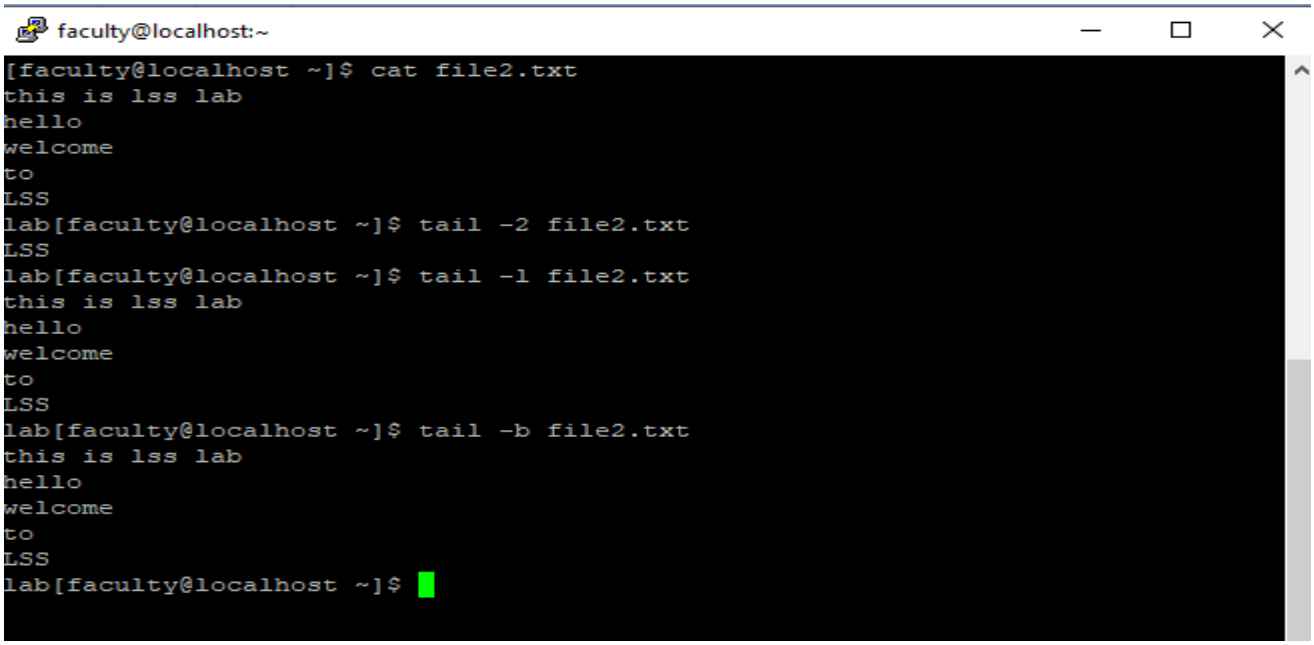
+N	skip N-1 lines
-N	N lines from end
-l	measured by lines(default)
-c	measured by characters
-b	measured by blocks
-r	output in reverse order

Example:

$ tail -2 file2.txt

$ tail –l file2.txt

$ tail -b file2.txt



e) Filters: cmp,comm.,diff,sort,uniq

cmp(compare) command: It examines two files byte by byte.the action it take depends on thye options used.

Syntax:

cmp options inputfiles

options:

-l list

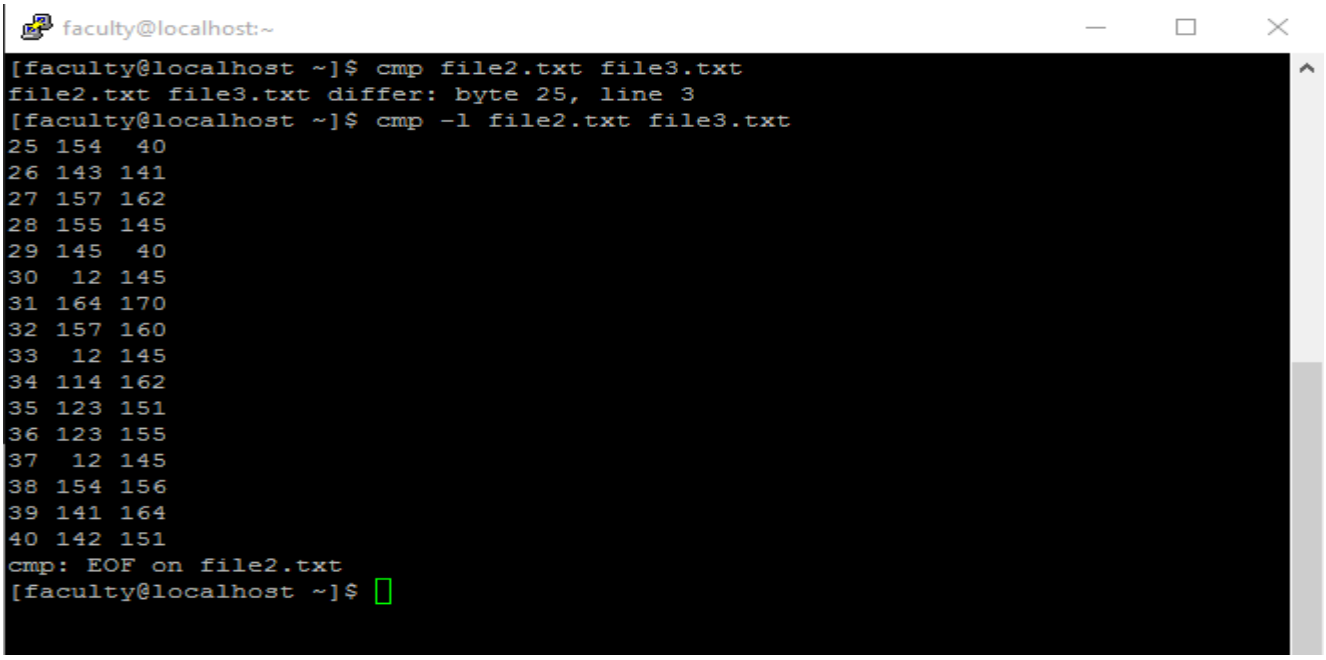
-s suppress list

Example:

$ cmp file2.txt file3.txt

$ cmp –l file2.txt file3.txt

$ cmp –s file2.txt file3.txt



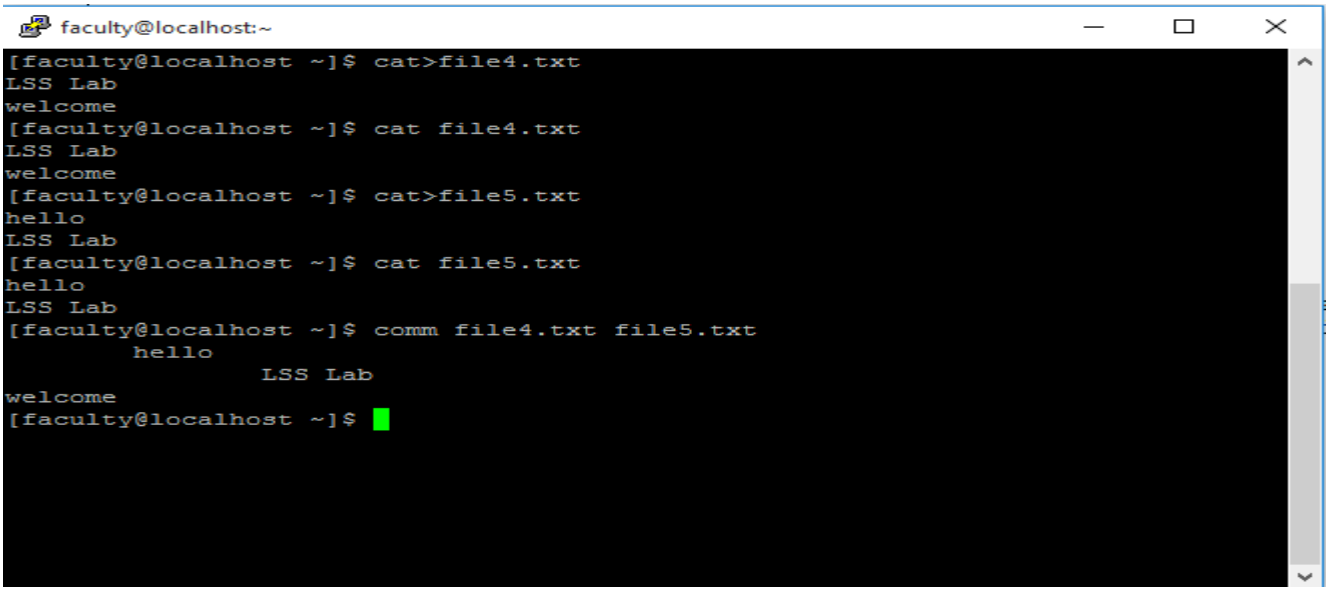
comm(common) command: It finds lines that are identical in two files.It compares the files line by line and displays the result in three columns.the left column contains unique lines in file1,the center column contains unique lines in file2,the right column contains lines found in both files.

Syntax:

comm inputfiles

Example:

comm file4.txt file5.txt



diff(difference) command:It shows line by line differences between two files.The first file is compared to the second file.The differences are identified such that the first file could be modified to make it match the second file.The diff command always works on files.

Syntax:

diff options [files or directories]

options:

-b ignore trailing blanks

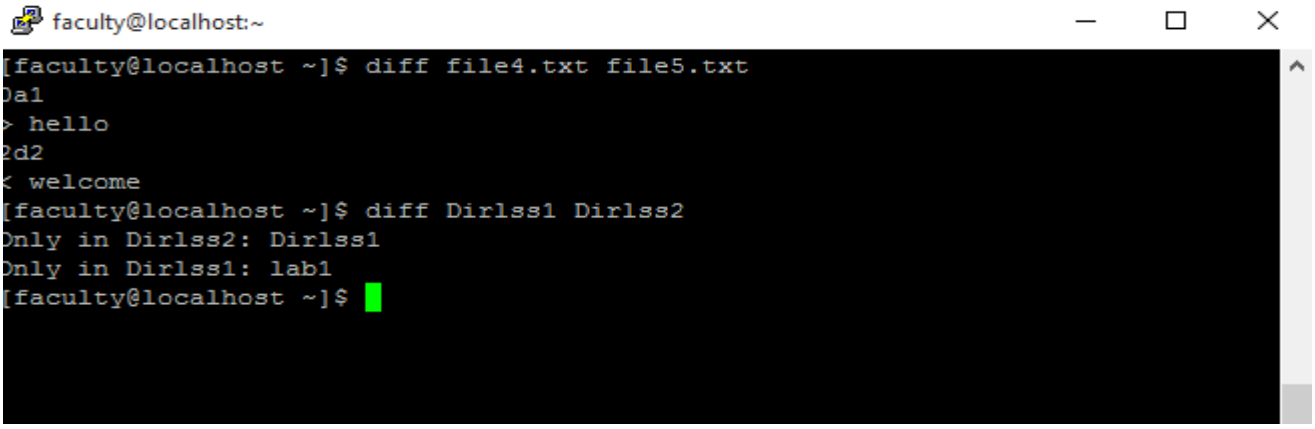
-w ignore whitespace

-i ignore case

Example:

$ diff file4.txt file5.txt

$ diff Dirlss1 Dirlss2



sort(sorting) command: It sorts the data lines in a text file using standard sorting rules.It interprets numbers as characters and performs a standard character sort,producing output that might not be what you want.By using –n parameter the sort command recognize numbers as numbers instead of characters,and sort them based on their numerical values.

Syntax:

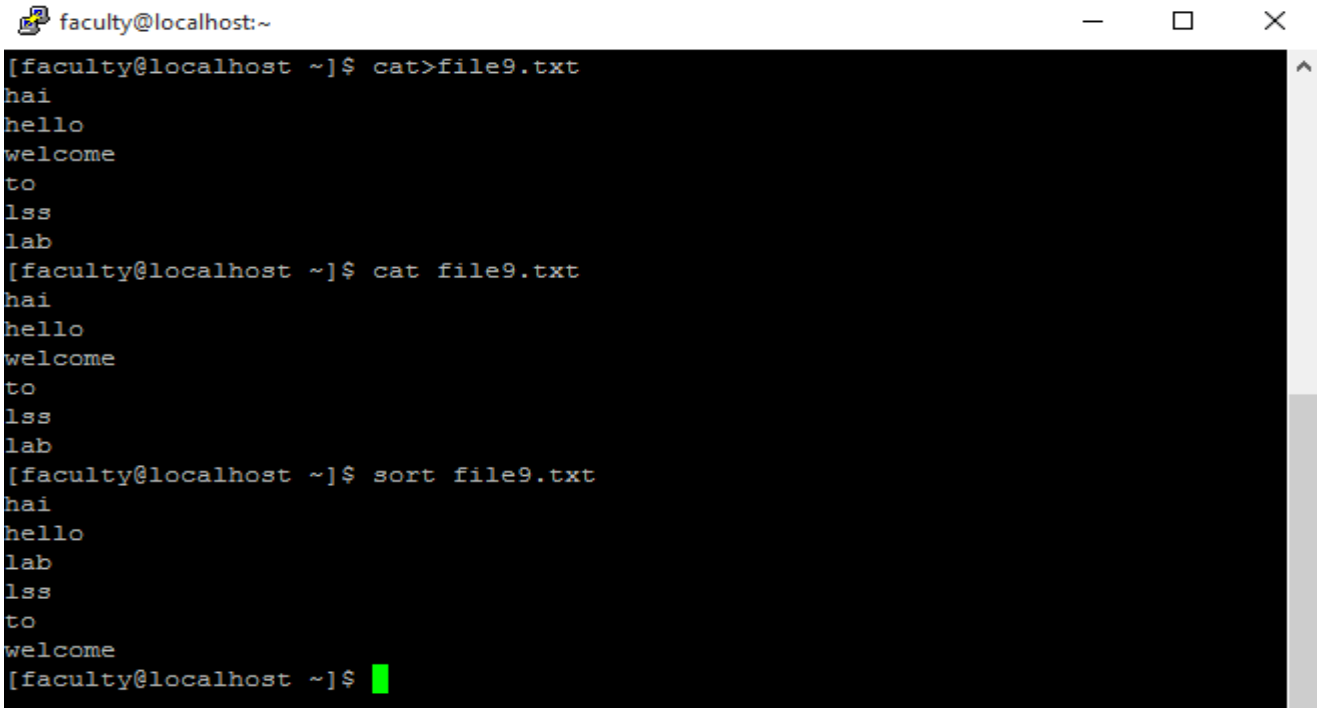
sort options inputfiles

options:

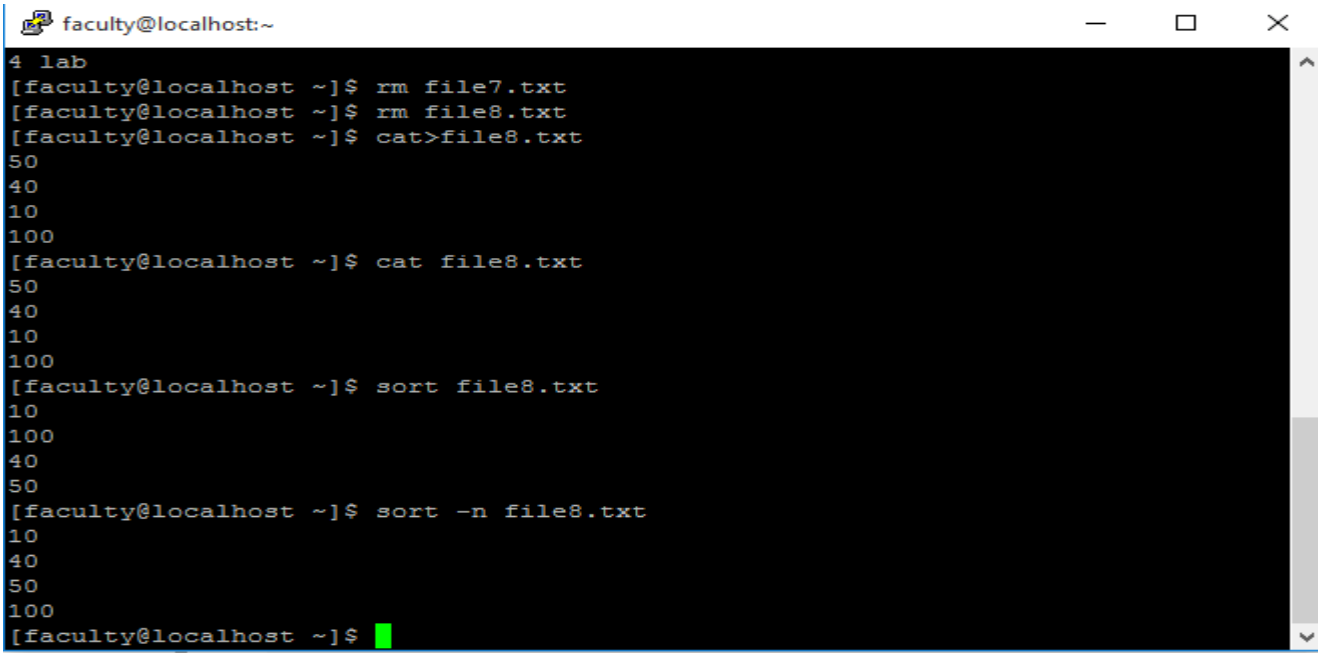
-n	recognizes numbers as numbers instead of characters.
-b	ignore leading blocks
-C	don't sort,but don't report if data is out of sort order
-c	don't sort,but check if the input data is already sorted.report if not sorted
-d	dictionary order
-f	ignore case
-g	general numeric sort
-i	ignore nonprinting
-M	month sort
-o	output file
-r	reverse
-u	unique
-z	zero terminated
-m	merge two already sorted data files

Example:

$ sort file9.txt



$ sort –n file8.txt



uniq(unique) command: It deletes duplicate lines,keeping the first and deleting the others.To be deleted,the lines must be adjacent.Duplicate lines that are not adjacent are deleted when the file is in sorted order.

Syntax:

uniq options inputfile

options:

-u only nonrepeated lines output

-d only repeated lines output

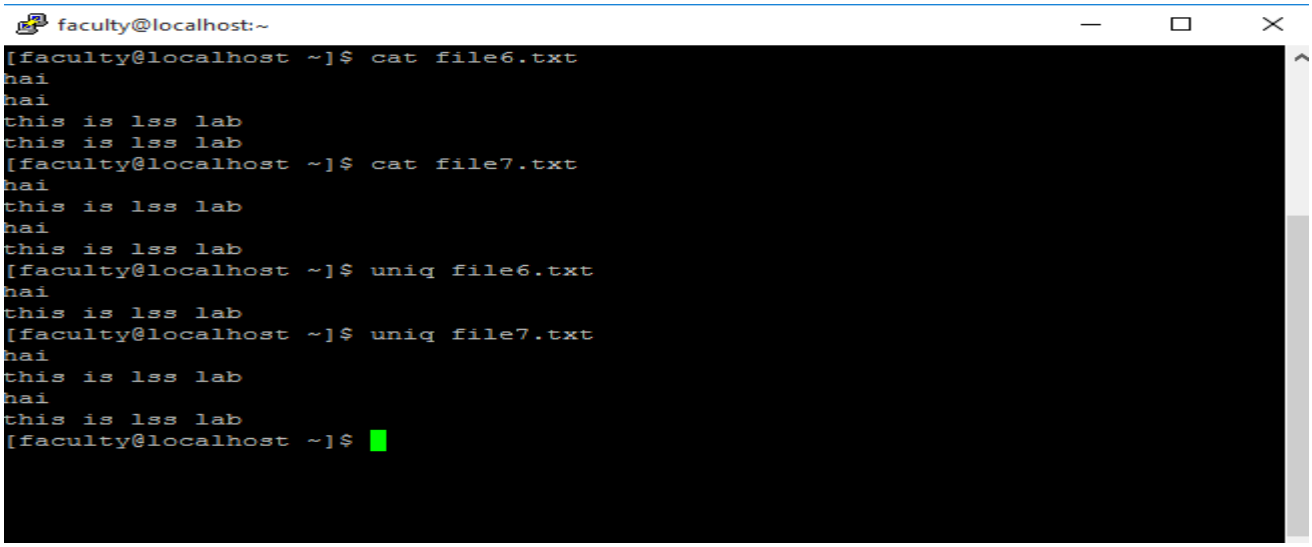
-c add count to lines

-f field

-s character

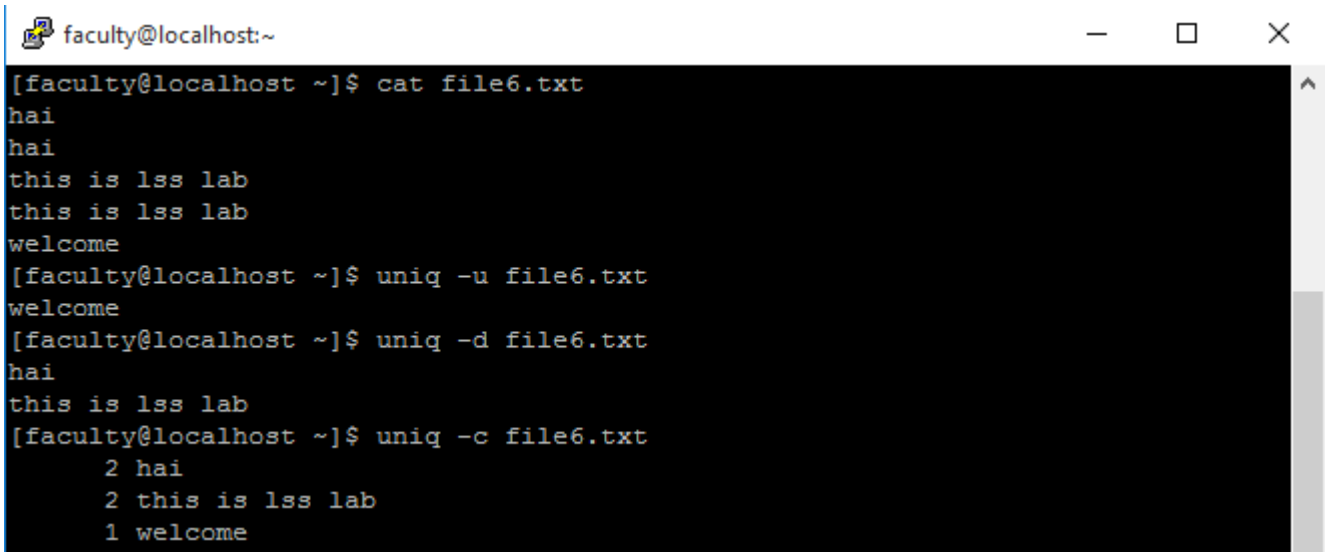
Example:

$ uniq file6.txt



$ uniq –u file6.txt

$ uniq –d file6.txt



f) Disk Utilities: du,df

du(disk usage) command: It is used to find disk usage for files and directories.By default du command displays all of the files,directories and subdirectories under the current directory,and it shows how many disk blocks each file or directory takes.

Syntax:

du options inputfile

options:

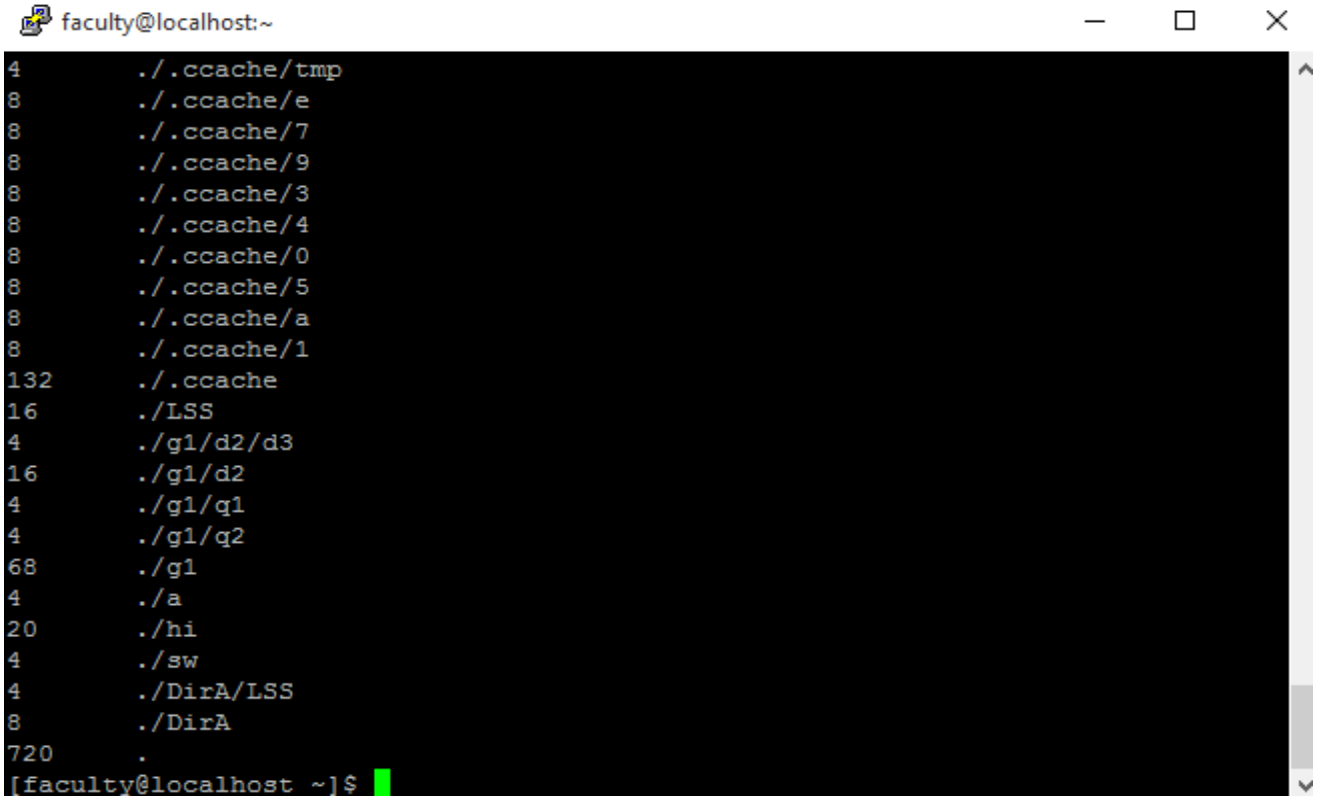
-c produce a grand total of all the files listed

-h print size in human readable form,using K for kilobyte,M for megabyte,G for gigabyte

-s summarize each argument

Example:

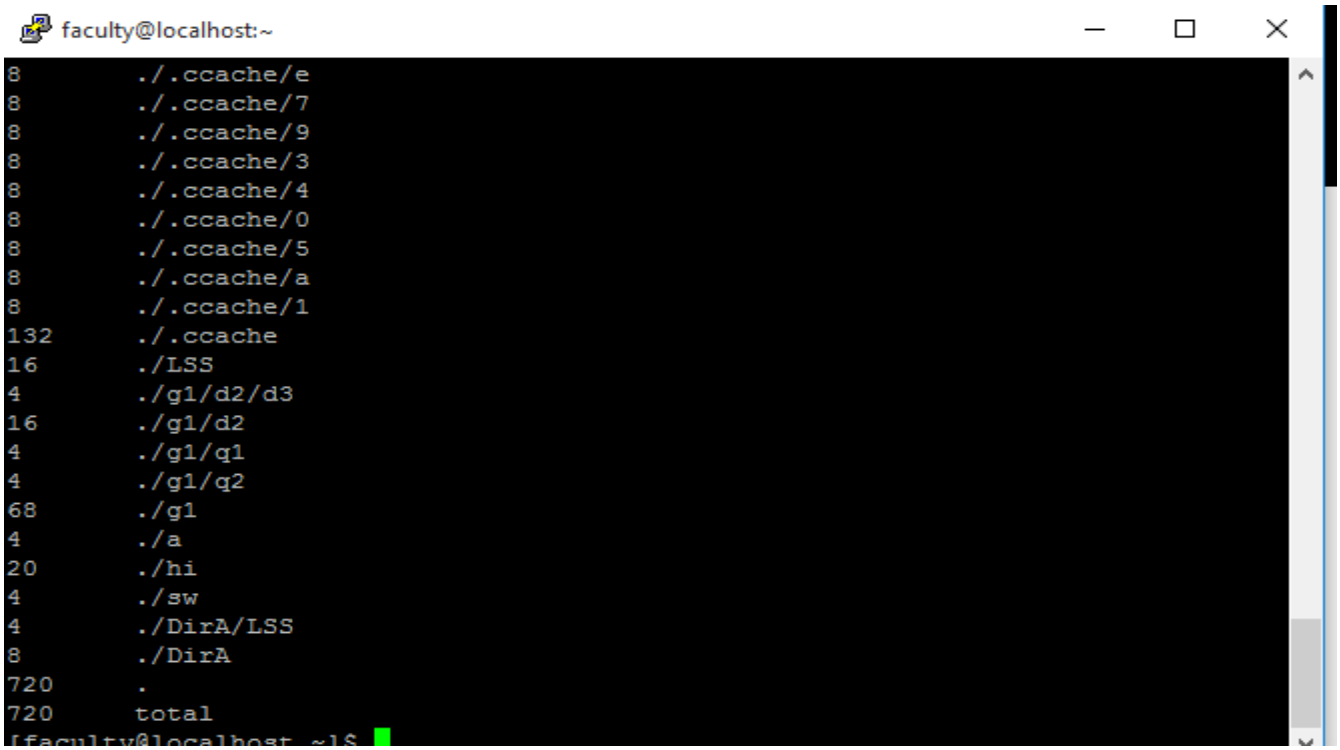
$ du



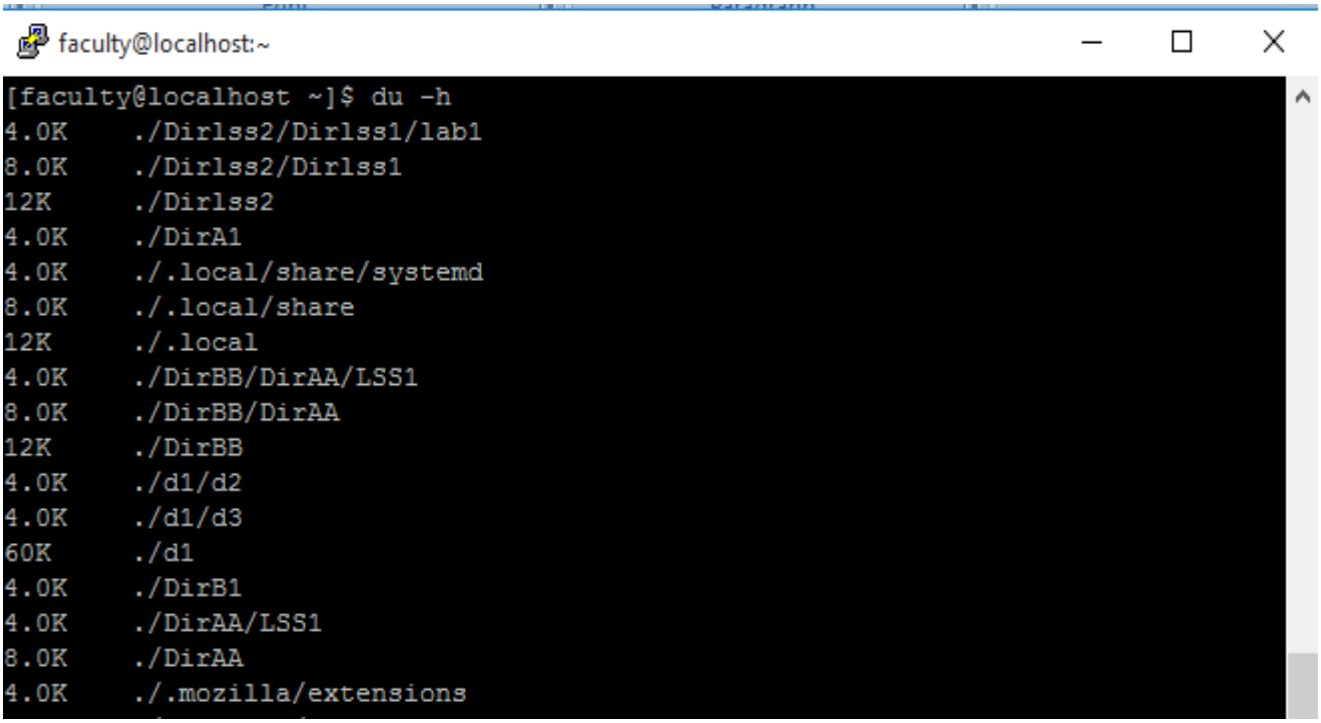
$ du –s


faculty@localhost:~
[faculty@localhost ~]$ du -s
720 .
[faculty@localhost ~]$

$ du –c


faculty@localhost:~
8 ./.ccache/e
8 ./.ccache/7
8 ./.ccache/9
8 ./.ccache/3
8 ./.ccache/4
8 ./.ccache/0
8 ./.ccache/5
8 ./.ccache/a
8 ./.ccache/1
132 ./.ccache
16 ./LSS
4 ./g1/d2/d3
16 ./g1/d2
4 ./g1/q1
4 ./g1/q2
68 ./g1
4 ./a
20 ./hi
4 ./sw
4 ./DirA/LSS
8 ./DirA
720 .
720 total

$ du –h



df(disk filesystem) command: It is used to show disk utilization for a system or disk space available on an individual device.

Syntax:

df options inputfile

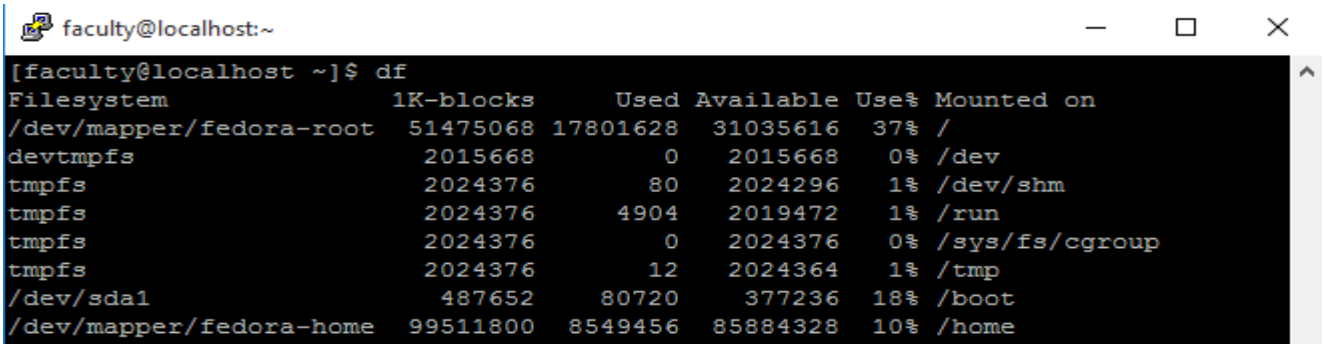
options:

-h information in human readable format

-hT information of particular partition in human readable format

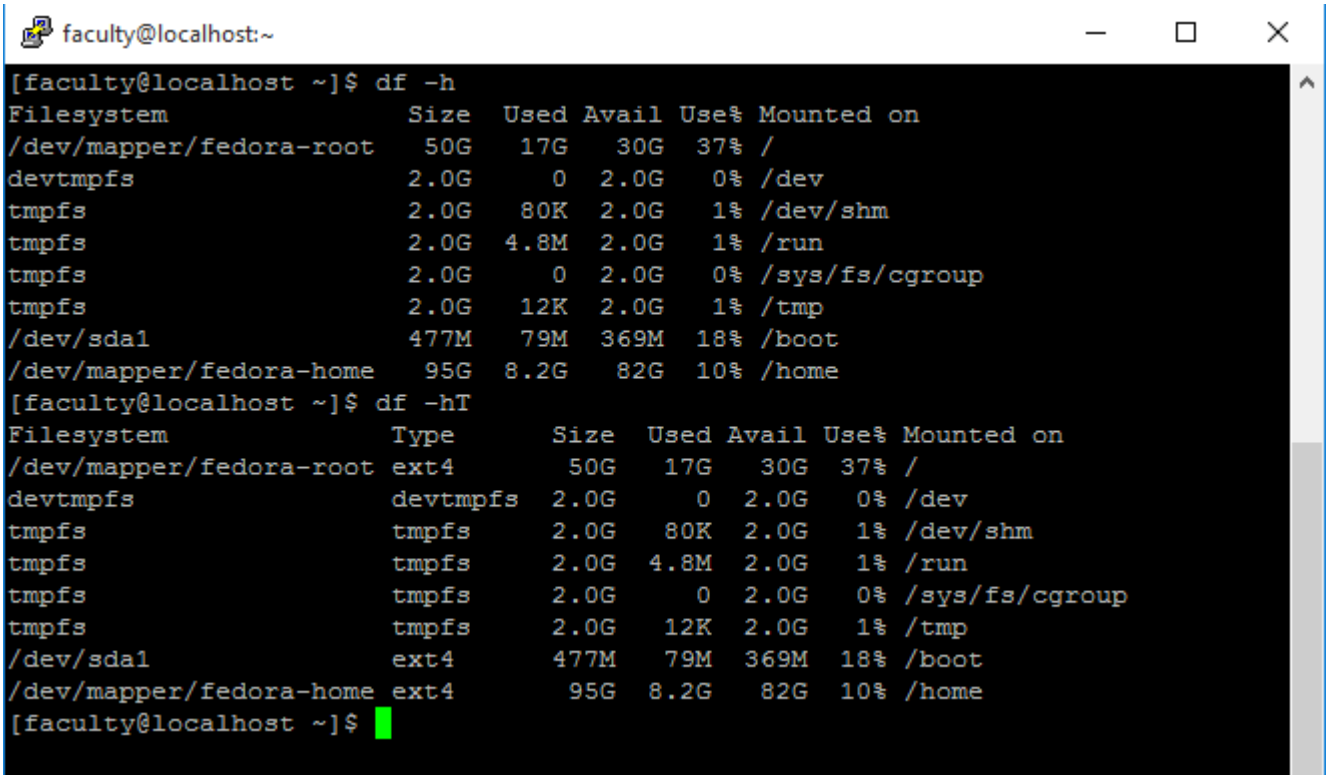
Example:

$ df



$ df –h

$ df -hT



PROGRAM -2 **Date:**

Aim: Develop a Shell Program to Display all the words which are entered as command line arguments.

Overview of Unix Command Line Arguments:

The Unix shell is used to run commands, and it allows users to pass run time arguments to these commands.These arguments, also known as command line parameters, that allows the users to either control the flow of the command or to specify the input data for the command.

How to write,compile and run shell programs:

Step1: By using vi command create shell script/program with .sh extension.

$vi filename.sh

Step2: In the vi editor type the program and for saving & quiting from vi, press **Esc :wq** then press enter

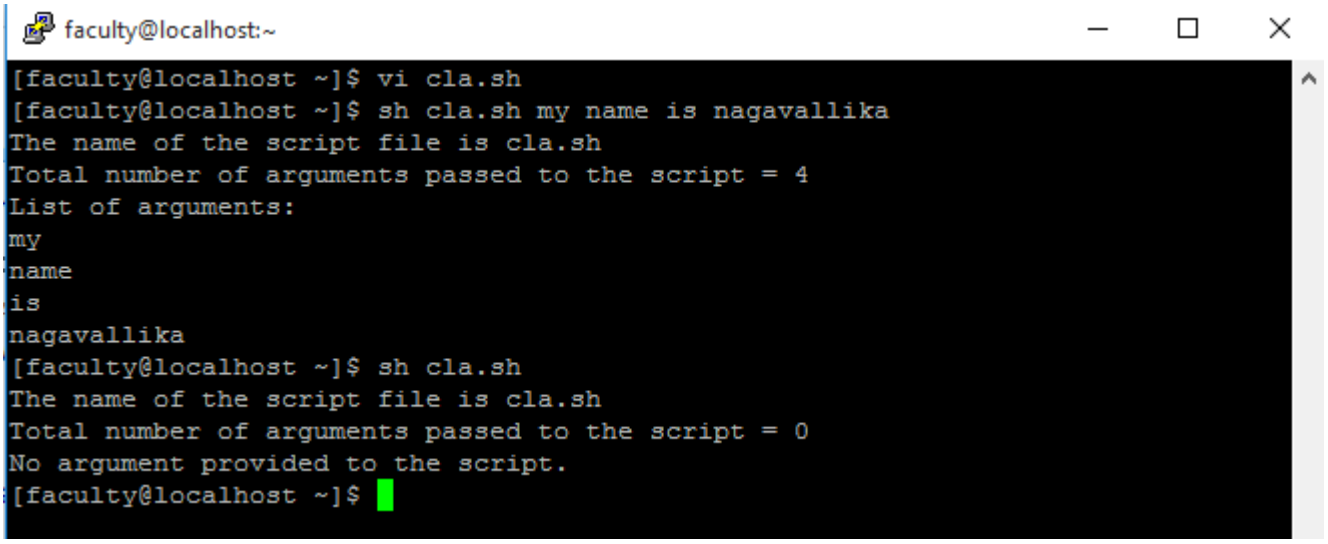
Step3: Compile and execute the shell script using following command

$sh filename.sh

Shell Script:

```
echo "The name of the script file is $0"
echo "Total number of arguments passed to the script = $#"
if [ $# -gt 0 ]
then
  echo "List of arguments:"
  for arg in $@
  do
    echo "$arg"
  done
else
  echo "No argument provided to the script"
fi
```

Output:



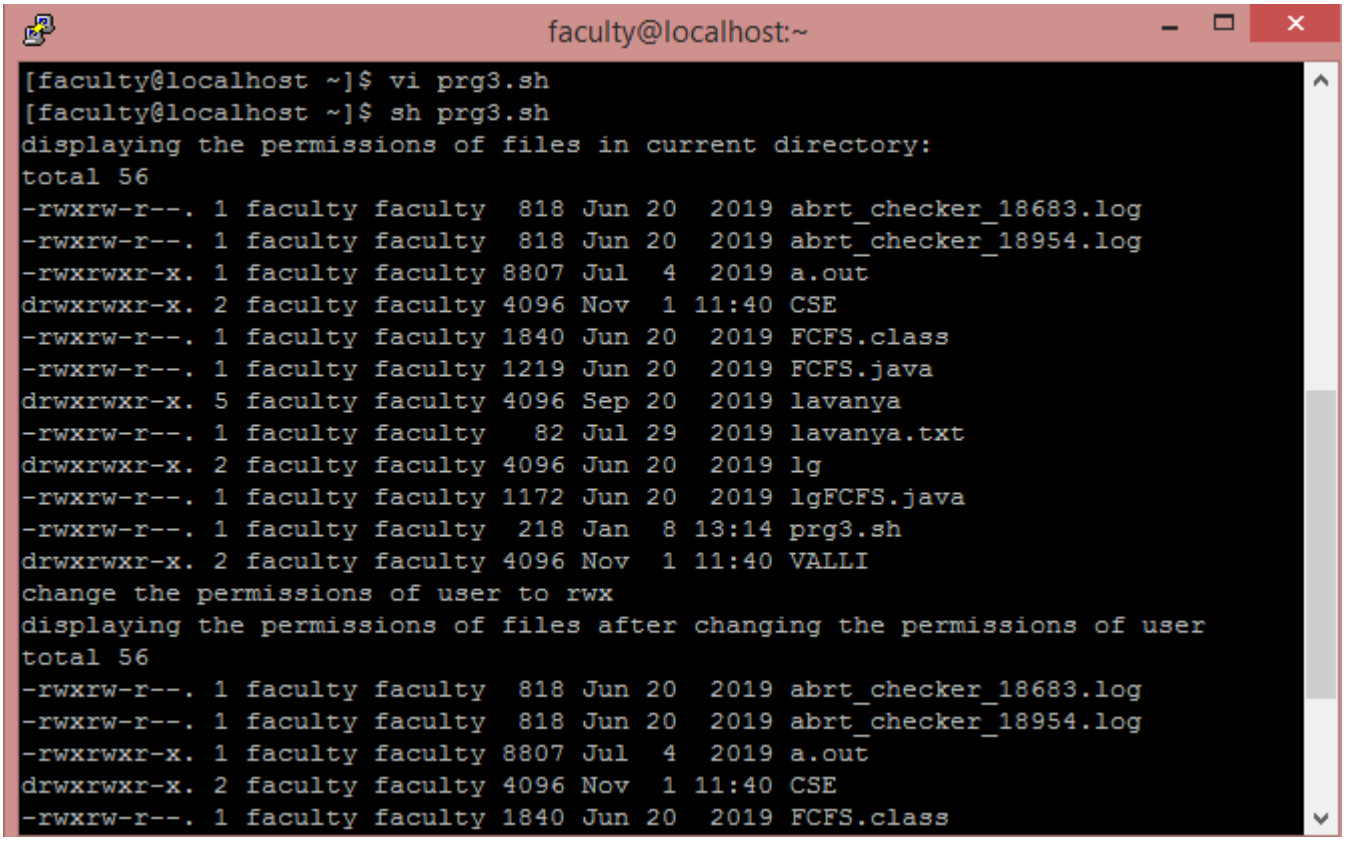
PROGRAM -3 **Date:**

Aim: Develop a shell script that Changes Permissions of files in PWD as rwx for users.

Shell Script:

```
echo "displaying the permissions of files in current directory:"
ls -l
echo "change the permissions of user to rwx"
chmod u=rwx *
echo "displaying the permissions of files after changing the permissions of  user"
ls -l
```

Output:



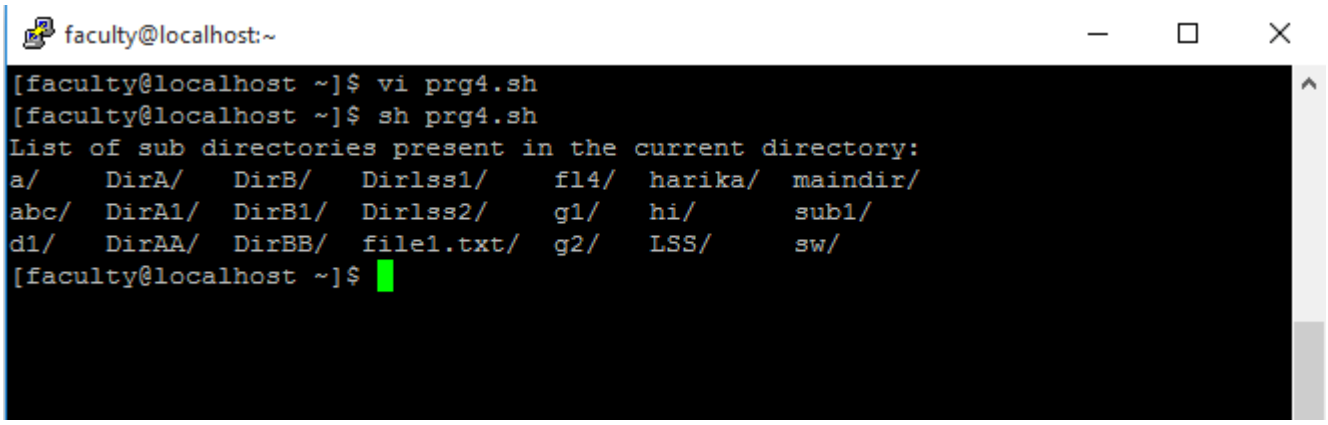
PROGRAM -4 **Date:**

Aim: Develop a shell script to print the list of all sub directories in the current directory.

Shell Script:

```
echo "List of sub directories present in the current directory  :"
ls -d */
```

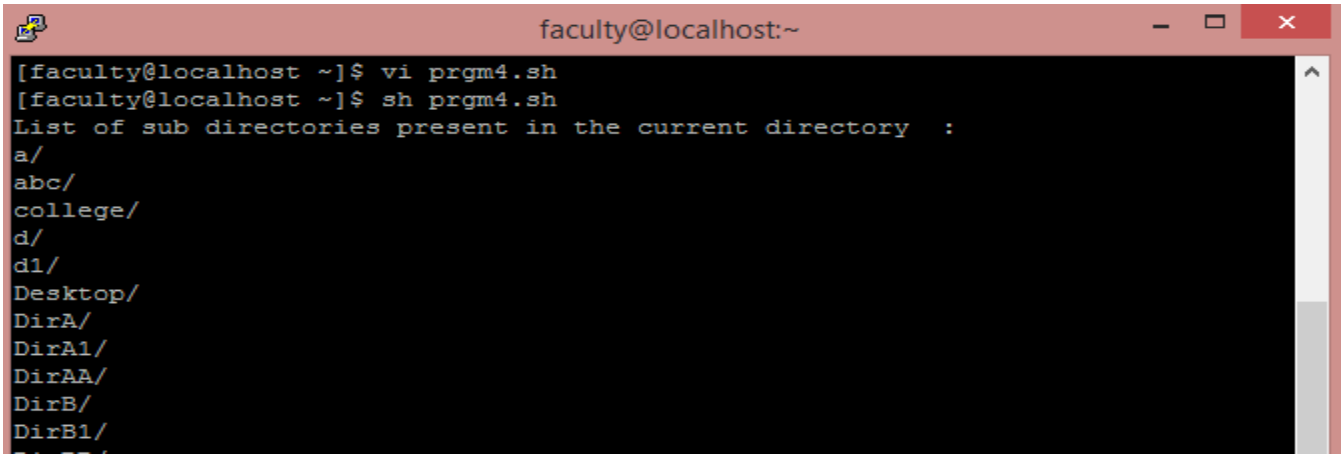
Output:



Shell script:

```
echo "List of sub directories present in the current directory  :"
for d in */
do
echo $d
done
```

Output:



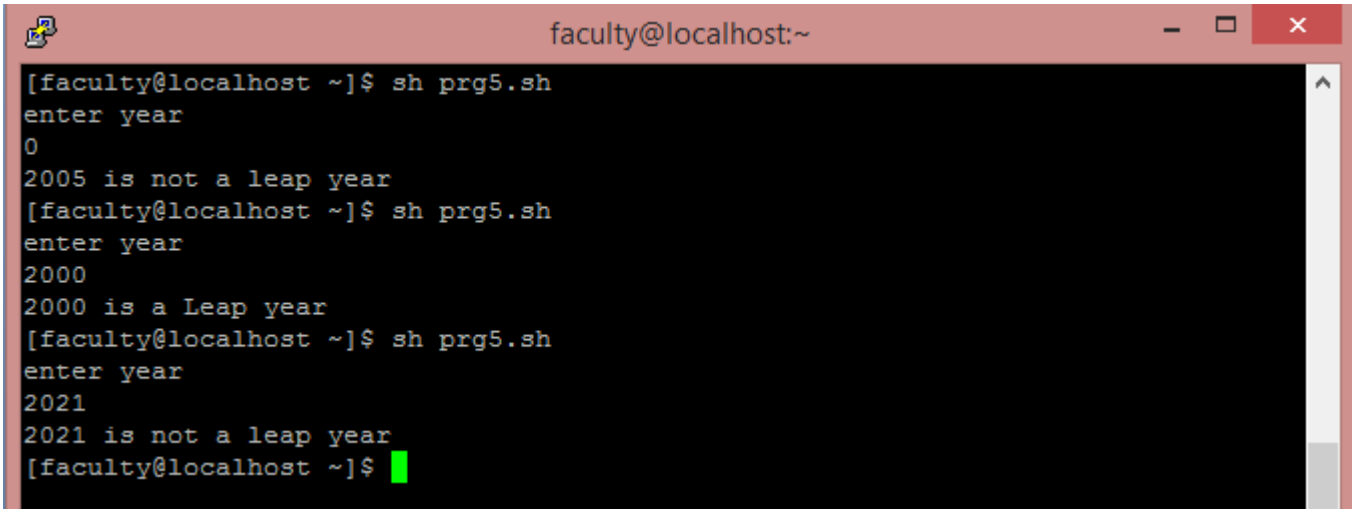
PROGRAM -5 **Date:**

Aim: Develop a Shell Program which receives any year from the keyboard and determine whether the year is leap year or not. If no argument is supplied the current year should be assumed.

Shell Script:

```
echo "enter year "
read year
#if year is less than or equal to 0 then current year is assigned to year variable
if [ $year -le 0 ]
then
year=`date | cut -d " " -f6`
fi
if [ `expr $year % 400` -eq 0 ]
 then
echo "$year is a Leap year"
elif [ `expr $year % 100` -eq 0 ]
 then
echo "$year is not a Leap year"
elif [ `expr $year % 4 ` -eq 0 ]
 then
echo "$year is  a Leap year"
else
echo "$year is not a Leap year"
fi
```

Output:



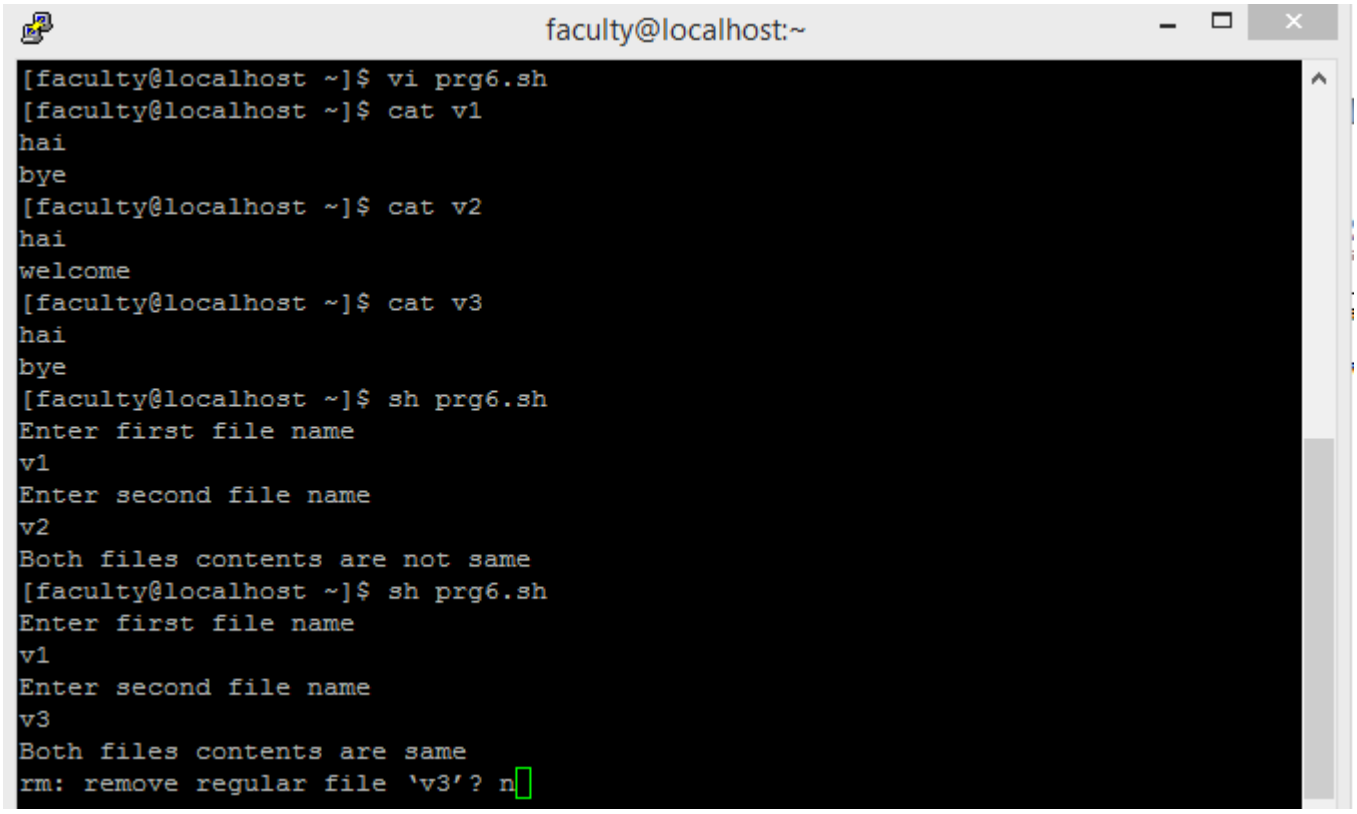
PROGRAM -6 **Date:**

Aim: Develop a shell script which takes two file names as arguments-If their contents are same then delete the second file.

Shell script:

```
echo "Enter first file name "
read file1
echo "Enter second file name "
read file2
if cmp -s "$file1" "$file2"
then
        echo "Both files contents are same"
        rm -i "$file2"
else
        echo "Both files contents are not same"
fi
```

Output:



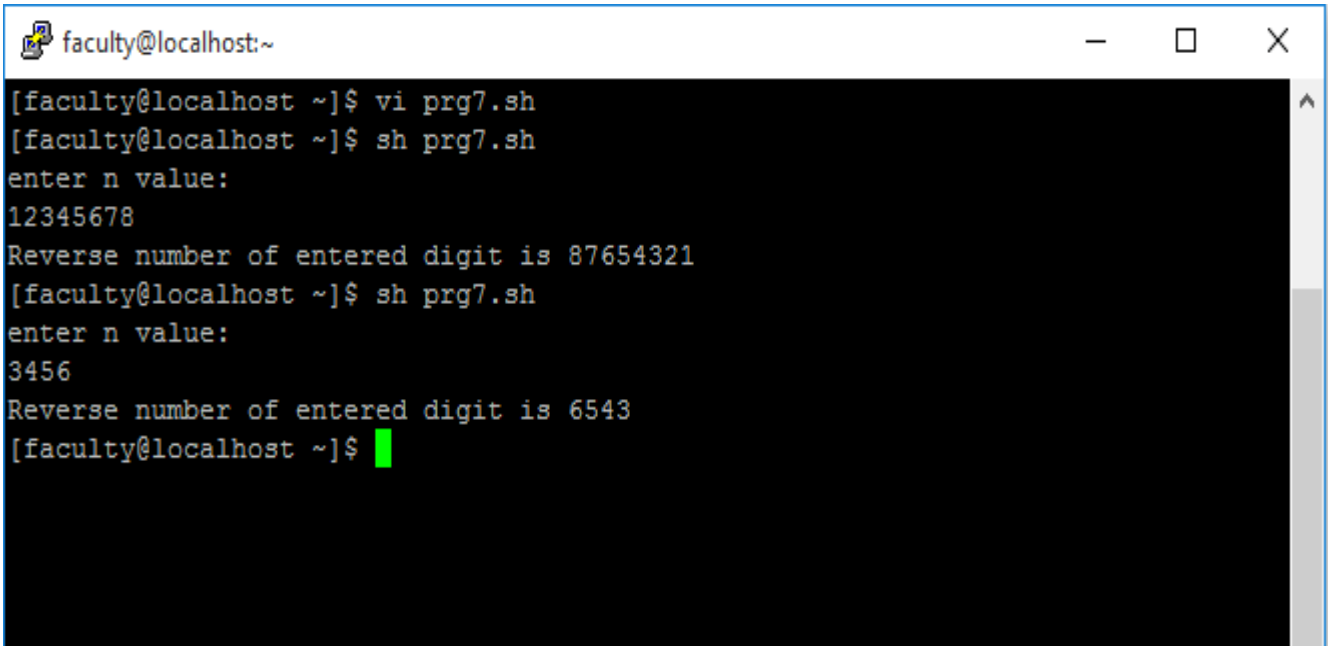
PROGRAM -7 **Date:**

Aim: Develop a shell script to print the given number in the reversed order.

Shell script:

```
echo “enter n value:”
read n
sd=0
rev=0
while [ $n -gt 0 ]
do
   sd=$(( $n % 10 ))
   rev=$(( $rev * 10 + $sd ))
   n=$(( $n / 10 ))
done
echo "Reverse number of entered digit is $rev:"
```

Output:



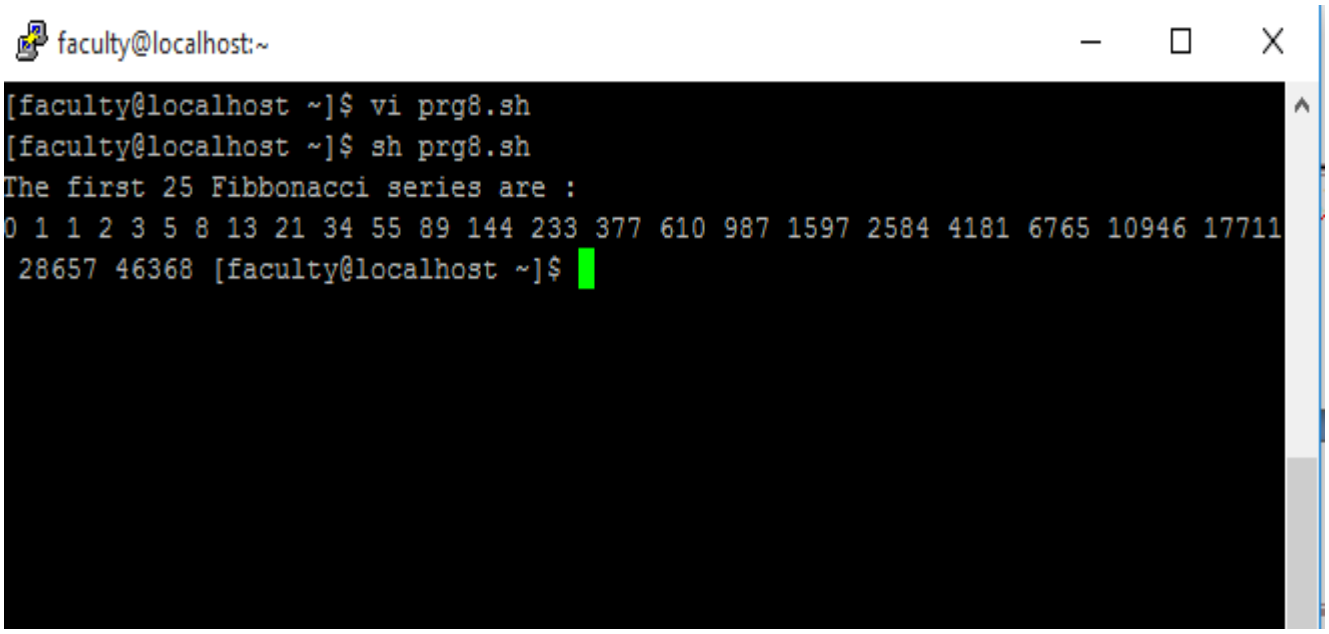
PROGRAM -8 **Date:**

Aim: Develop a shell script to print first 25 Fibbonacci numbers.

Shell script:

```
n=25
a=0
b=1
echo "The first 25 Fibbonacci series are : "
for((i=0;i<n i++))
do
   echo -n "$a "
   fn=$((a + b))
   a=$b
   b=$fn
done
```

Output:



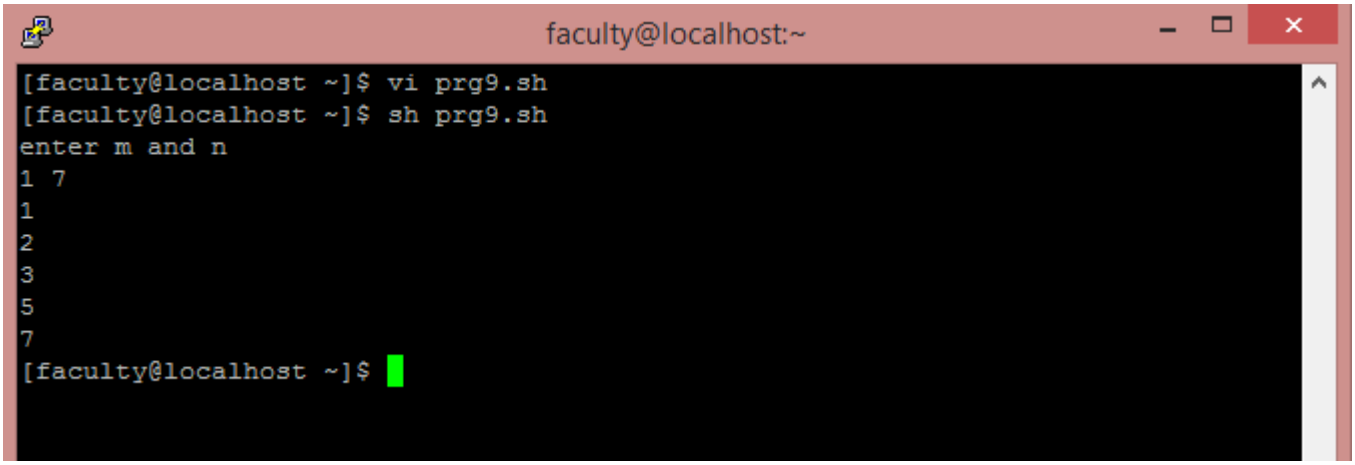
PROGRAM -9 **Date:**

Aim: Develop a shell script to print the Prime numbers between the specified range.

Shell script:

```
echo "enter m and n"
read m n
for a in $(seq $m $n)
do
        k=0
        for i in $(seq 2 $(expr $a - 1))
        do
                if [ $(expr $a % $i) -eq 0 ]
                then
                        k=1
                        break
                fi
        done
        if [ $k –eq 0 ]
        then
        echo $a
        fi
done
```

Output:



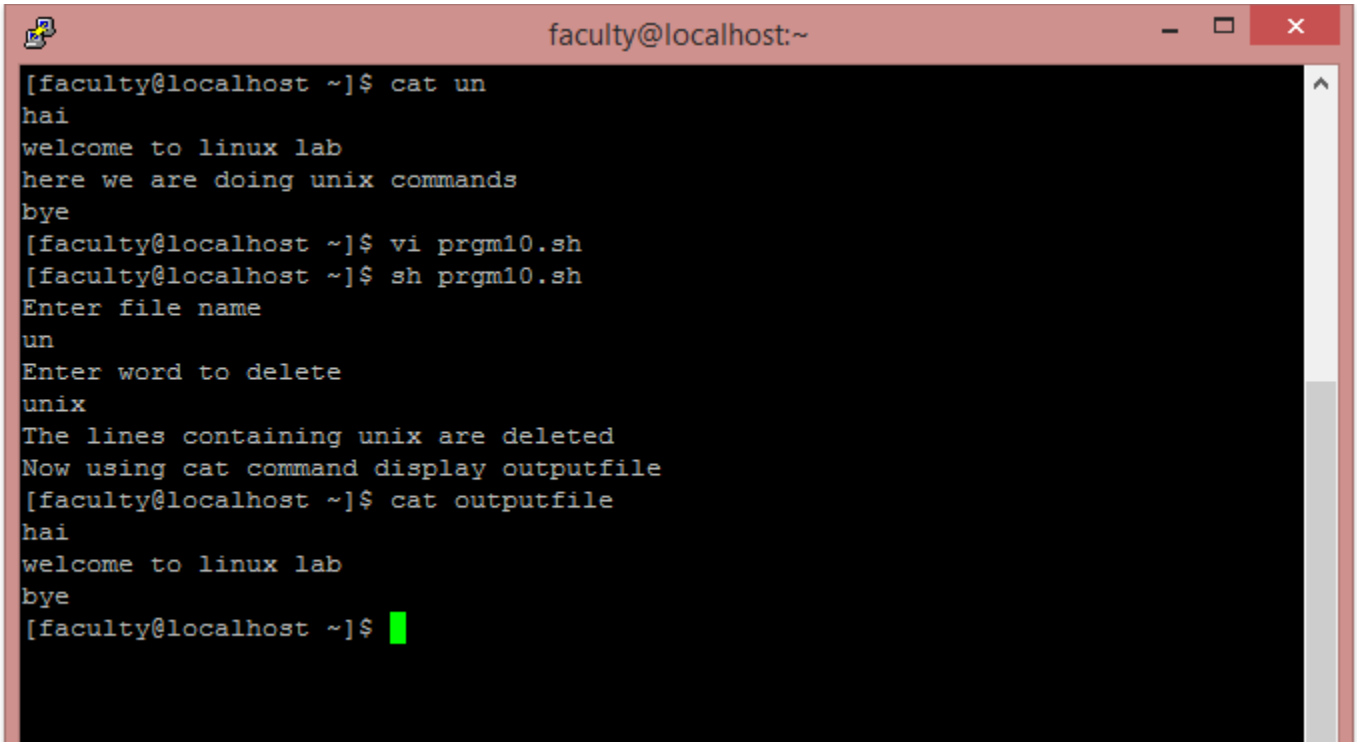
PROGRAM -10 **Date:**

Aim: Develop a shell script to delete all lines containing the word ‘unix’ in the files supplied as arguments.

Shell script:

```
echo "enter file name"
read FILE
echo "Enter word to delete"
read word1
sed -e '/unix/d' $FILE>outputfile
echo "The lines containing unix $word1 are deleted"
echo "now using cat display outputfile"
```

Output:



PROGRAM -11 **Date:**

Aim: Develop a shell script Menu driven program which has the following options.

i) contents of /etc/passwd

ii) list of users who have currently logged in.

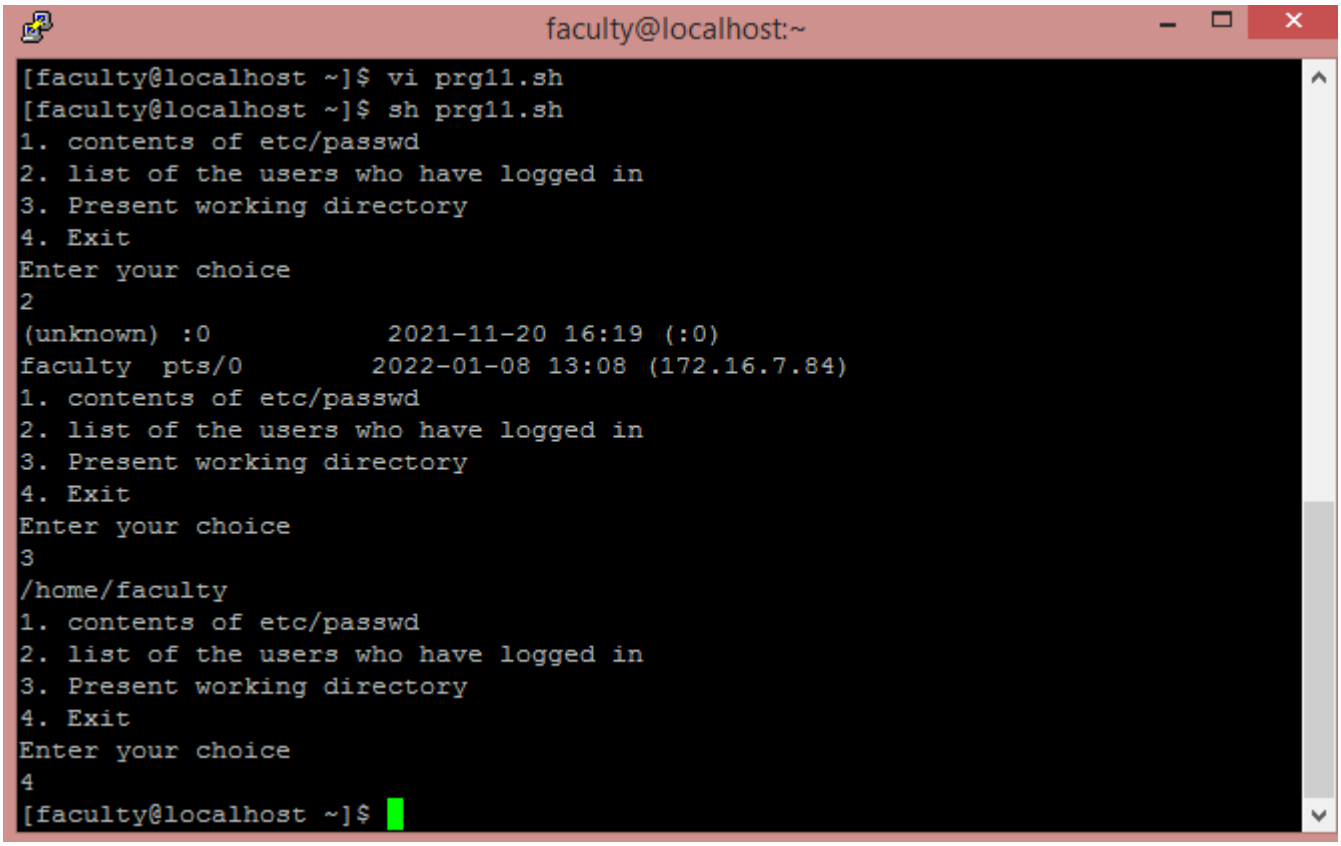
iii) present working directory.

iv) exit.

Shell script:

```
choice=0
while [ $choice -lt 4 ]
do
tput clear
echo "1. contents of etc/passwd"
echo "2. list of the users who have logged in"
echo "3. Present working directory"
echo "4. Exit"
echo "Enter your choice "
read choice
if [ $choice -eq 1 ]
then
cat /etc/passwd
elif [ $choice -eq 2 ]
then
who
elif [ $choice -eq 3 ]
then
pwd
fi
done
```

Output:


faculty@localhost:~
[faculty@localhost ~]$ vi prg11.sh
[faculty@localhost ~]$ sh prg11.sh
1. contents of etc/passwd
2. list of the users who have logged in
3. Present working directory
4. Exit
Enter your choice
2
(unknown) :0 2021-11-20 16:19 (:0)
faculty pts/0 2022-01-08 13:08 (172.16.7.84)
1. contents of etc/passwd
2. list of the users who have logged in
3. Present working directory
4. Exit
Enter your choice
3
/home/faculty
1. contents of etc/passwd
2. list of the users who have logged in
3. Present working directory
4. Exit
Enter your choice
4
[faculty@localhost ~]$