```
#!/bin/bash
# author : Surya Dev Singh Jamwal
# course : PG-DBDA
# user ID : sdevsinx
# date created : 2023 Mar 20
# title : Linux module day02
#
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++


# 1. grep command
# search for the pattern in a file
# grep command stands for Global Regular Expression Print
# It'll check the entire text file for a pattern

grep "hello" check.txt      # searches for "hello"
grep -n "o" check..txt      # shows line number for pattern
grep -c "to" check.txt      # shows the count of pattern in file
grep -v "way" check.txt     # shows the line which did not match the
pattern
grep -e "to" check.txt      # you can specify multiple search expressions
using '-e'
grep -r "he" /home/sdevsinx # searches recursively in dir & its sub-dirs
for the specified pattern
grep -i "TO" check.txt      # ignores case while matching pattern

vi check.txt
mkdir t
cd t/
vi t.txt
mkdir t1
vi t1.txt
cd t1
vi t11.txt
cd ..

# '-n' shows line number for the searched pattern
# '-c' shows the count for the searched pattern
# '-v' shows the lines which did not match the pattern
```

```
# '-v' used to specify a pattern/ multiple patterns to match
# '-r' used to search recursively in dir & its sub-dirs
# '-i' to ignore case while matching pattern


# 2. find
# it is used to search specific string of characters
find check.txt      # to find the file with specified name
find ./t t.txt      # to find the file in t/ dir
find / t.txt        # to find the file in / (root) dir / all of the file
system
find ~ t.txt        # to find the file in ~ (home) dir
find . t.txt        # to find the file in . (current) dir
find . -name "check.txt"    # to find a file with specified name
find . -name "*.txt"    # to find a file name with extension '.txt'
find . -type d      # to find all dirs
find . -type f -size +1M    # searches for file with '-type f', & searches
for size more than 1Mb with '-size +1M'
find . -iname "CHECK.txt"   # searches for file with specified name, but
with ignoring case
find . -name "*.ext" delete # searches for file with specified
name/extension, and deletes it

# 3. difference between grep & find
```

grep	find
grep is used to find a pattern inside a file (contents of file)	find is used to find a file based on their name, type, size and permission

```


# 4. locate [used in big data]
# Locate command is used to find files & dirs
locate check.txt
locate dir_name
locate -r dir_path  # to locate file
locate -r '\*.txt'  # to locate files with extension '.txt' recursively
locate -u cdac      # to locate files owned by specified user

locate -e check.txt # to locate the updated files
```

```
# 5. Difference between find & locate
```

find	locate
find is used to find a file based on their name, type, size & permission	the search in locate is done according to previously created database of files & dirs
find is slower	locate is faster than find
files once deleted can't be found using find command	files once deleted can be found even after it has been deleted

```
# 6. sort
# This command is used to sort file contents
sort  test1.txt     # to sort lines in file A-Z
sort -r test1.txt   # to sort lines in file Z-A (reverse sort)
sort -n test2.txt   # to sort numerically
sort -k 5 test1.txt # to sort on a particular line
sort -u test3.txt   # deletes all duplicate lines and then sorts

# 7. uniq
# this command removes adjacent duplicate lines in a file
uniq uni.txt    # removes adjacent duplicate lines, but prints duplicates
only once
uniq -c uni.txt # counts occurrences of adjacent duplicate lines
uniq -u uni.txt # returns only unique lines
uniq -d uni.txt # returns only duplicate lines


# 8. difference between sort & uniq
```

uniq	sort
uniq removes all the duplicate adjacent lines	sorts lines in file in A-Z
uniq requires file to be sorted	sort takes unsorted file as input
uniq doesn't need any other command to remove duplicates from a file	sort command is combination of commands like uniq, cut to manipulate & analyze data

```
# 9. kill
# to kill any running processes
```

```
wait &  # launches wait in background with job id 1
ps  # to get all running processes
kill -9 2292    # kills  with sigterm 9

# PID   # process ID
# TTY   # terminal time
# TIME  # estimated time of CPU execution
# CMD   # command that has been launched

#10. ps
# to check processes
ps -a   # returns only processes running in other terminals
ps -e   # returns all processes associated with current terminal as well
as other terminals
ps -r   # returns only running processes
ps -x   # returns status of a particular process


#11. cut
# this command is used to extract specific columns or fields form a file
# This helps in manipulating the text data
# '-d'  specifies delimiter such as ',','.', etc
# '-f'  specifies field / columns which you want to extract
# '-f1,2'   specifies field 1 & 2
# '-f1-2'   specifies field 1 to 2
vi  cu.txt
cut -d ',' -f1 cu.txt   # '-d' specifies delimiter as ',' , '-f' specifies
field number 1
cut -d ',' -f2 cu.txt   # '-d' specifies delimiter as ',' , '-f' specifies
field number 2
cut -d ',' -f3 cu.txt   # '-d' specifies delimiter as ',' , '-f' specifies
field number 3
cut -d ',' -f1,2 cu.txt # '-d' specifies delimiter as ',' , '-f' specifies
field number 1 & 2
cut -d ',' -f1-2 cu.txt # '-d' specifies delimiter as ',' , '-f' specifies
field range 1 to 2

#12. links
# links work as pointer to file
```

```
# when we're creating a link, we create a shortcut to access that parent
file
# changes are reflected in both hard & soft link, when parent file is
modified
#
# Two  types of links:
#  a. Hard links (link still contains data even if parent file is deleted)
#  b. soft links (link gets broken when parent is deleted)
#
#  a.HARD LINK
# in hard link, link exists there even if the parent file is
deleted(contents are also present)
   ln file_name link_name   # syntax
vi li.txt           # create parent file
ln li.txt hlink.txt     # creates hard link from parent file without '-s'
echo "cdac mumbai" > li.txt # make changes in parent file
cat li.txt          # check changes in parent file
cat hlink.txt           # check changes in link
rm li.txt           # removes parent file, but link file still stays
cat hlink.txt           # still prints data from link file, whereas parent
file is already deleted
#
#   b.SOFT LINK
# soft link can also be called as symbolic link
# This kind of links cannot be updated  if the main link is deleted/its
path is changed
    ln -s file_name link_name   #syntax
vi sli.txt          #  create a parent file
ln -s -v sli.txt  slink.txt # to create soft link using '-s'
cat sli.txt         # to print contents of hli.txt
cat  slink.txt          # to print contents of hlink.txt, same as parent
file
echo "cdac mumbai" > sli.txt    # changing contents of parent file
cat sli.txt         # to print changed contents of parent file
cat slink.txt           # to print changed contents of soft link too
ls -l slink.txt         # to get details of soft link
rm sli.txt          # to remove parent file
cat slink.txt           # to print contents of soft link, but throws error
as link is broken
```

```
#13. to list users
cat /etc/passwd     # to list users
compgen -u      # to list users
getent passwd       # to list users

#14. add a user
adduser check1      # to add user 'check1", run with admin rights

#15. delete a user
userdel check1      # to delete user, run with admin rights
rm -r /home/user/cdac   #

#To-Do
GID UID
```