**Day 1: Linux Commands and filters**

**User Info:**

1. Logging in and Logging out

login

ssh user@host

CTRL+d

exit

logout

2. Changing user password

passwd

3. Basic user info commands

finger

finger [username]

4. Change user / finger information

chfn

5. Who are working on the system

users

who

w

6. Who am I?

who am i

whoami

id

**Date & Time:**

1. What is the date and time now?

date

2. What is the date in YYYY-MM-DD format?

date "+%Y-%m-%d"

3. What is the GMT time?

date -u

4. Show the calender of the current month

cal

5. Show the calender of last month, current month and next month

cal -3

6. Show the calender of a particular year

cal 2014

7. Try this, and tell us your observation about this year.

cal 1752

8. Tell us, the day on which you were born.

**Shell Features: alias, unalias, pipes, redirection, heredoc, \*, ?**

1. See the list of aliases

alias

2. Define an alias "f" for "finger" command

alias f=finger

3. Remove the alias "f"

unalias f

4. There is an utility called wc, which gives the word count.

How many users are logged in

who | wc -l

5. There is an utility 'seq' which can generate numbers.

Syntax: seq [[from=1] [step=1]] to

To generate 1 to 20 numbers

seq 1 20

To generate 1 to 10 with .5 increments

seq 1 .5 10

Put the output 'seq 1 20' command in numbers.txt file

seq 1 20 > numbers.txt

6. Check howmany lines are there in numbers.txt

wc -l < numbers.txt

Append the numbers.txt with numbers 21 to 40

seq 21 40 >> numbers.txt

7. How many multiples of 7 are there between 0 and 1000

seq 0 7 1000 | wc -l

8. Count the number of lines in the given text if it ends with the

marker END using heredoc

wc -l <<END

line1

line2

line3

END

**Filesystem: ls, cd, mkdir, touch, cp, mv, rm, ln, pwd, chmod, du, df,**

Filesystem:

1. Know about your current working directory

pwd

2. Make a directory "workshop"

mkdir workshop

3. List the contents of the current directory

ls

4. List in long format

ls -l

5. Show all the files(long list) including the hidden ones(the files

starting with a .)

ls -al

6. List the / direcotry sorted by the most recent modified first

ls -lt /

7. Show the list of files sorted by their size (biggest one first) of

/etc directory

ls -lS /etc

8. Change working directory to "workshop"

cd workshop

9. Create a directory to hold the notes and activities for day1, day2, day3 and having two subdirectories session1 and session2 for each day

mkdir -p day{1,2,3}/session{1,2}

10. List of files and directories recursively

ls -R

11. Remove the directories for day2 and session1 and session2

rmdir day2/session\*

rmdir day2

or

rm -r day2

rm -r day3

or

rm -r day[23]

or

rm -r day{2,3}

12. Create a zero size file "notes.txt" in day1/sesssion1 with touch

touch day1/session1/notes.txt

13. Check the list of files and directories

ls -lR \*

14. Copy day1 (recursively) into day2 and day3

cp -r day1 day2

cp -r day1 day3

15. Check the capacity, used, and available space on various filesystems

df

16. Check how much space is occupied by your home directory

du -s ~

17. Make a link to teacher-owned exercise sheet

ln ~teacher/LinuxWorkshop/day1/exercises.txt .

18. Check the acccess permissions, link count and inode number using ls

-li for exercises.txt file

ls -li exercises.txt

19. Copy exercises.txt file to exercises2.txt, that will be owned by you.

cp exercises.txt exercises2.txt

20. Attempt the change permissions on exercises.txt and exercises2.txt

chmod o-r exercises.txt exercises2.txt

or

chmod 0640 exercises.txt exercises2.txt

ls -li

21. Change to session1 of day1 directory

cd day1/session1

22. Revert back to parent directory i.e. day1

cd ..

23. Go to teachers home directory and list the files and folders

cd ~teacher

ls

24. Get back to your home directory

cd

or

cd ~

**Shell Builtins: pushd, popd, dirs, history, ulimit, umask**

1. Remeber your current working directory and change to your

day1/session2

pushd ~/day1/session2

2. Remeber your current working directory and change to teachers directory day1/session2

pushd ~teacher/day1/session2

3. Remember the current directory and change over to /tmp

pushd /tmp

4. Check your stacked working directories

dirs

5. Pop of one by one and revert to the previours working directories.

popd

dirs

popd

dirs

popd

6. Check your command history

history

7. Check the value of umask

umask

Create a file and direcotry and check the permissions

touch abc.txt

mkdir tmp

ls -ld abc.txt tmp

8. Change the umask to mask out the permissions to the group and others

umask 0077

Now create another file and directory

touch abc2.txt

mkdir tmp2

ls -ld abc2.txt tmp2

9. Find out the resource limits for processes for you

ulimit -a

10. Restrict CPU time to 1 second

ulmit -t 1

11. Now check the limits

ulimit –a

12. Execute a program which takes less than one second

time seq 1 1000 > /dev/null

13. Execute the same program with a differnt parameters to take more

than one second

time seq 1 1000000 > /dev/null

**Filters:**

**Filters: cat, tac, rev, split, nl, more, less, fold, head, tail, seq, yes, paste, cut, join, od, tr, sort, uniq, diff, comm, grep, echo, printf, look, tee, wc,**

1. concatenate files and print on the standard output

cat exercises.txt

2. Enter 3 multiples of 30 into mutiples.txt from keyboard

cat > multiples5.txt

30

60

90

CTRL+d

ls

3. Concatenate numbers.txt multiples.txt into morenums.txt

cat numbers.txt multiples.txt > morenums.txt

4. See the mutiples.txt in reverse order of lines

tac mutiples.txt

5. See the list of exercises in reverse order (note each exercise is

separated by blank line)

tac -r "\n\n" exercises.txt

6. Reversing line by line (the character of the line)

rev multiples.txt

7. Check the reversal of a reversal

rev mutilples.txt | rev

8. Similary check the tac of tac

6. Splitting a file into smaller pieces (100 lines for each).

Generate a file big.txt with a sequence 1050

seq 1 1050 > big.txt

split -d -l 100 big.txt part

7. Check each file line count

wc -l part\*

8. Reassemble the part files into bigcopy.txt

cat part\* > bigcopy.txt

9. Compare the original file and reassembled file

diff -sq big.txt bigcopy.txt

diff -sq big.txt part01

10. List the file multiples.txt content with linumbers

nl multiples.txt

11. See the contents of big.txt pagenated by more or less

more big.txt

less big.txt

cat big.txt | more

cat big.txt | less

12. Wrap the file exercises.txt to a width of 20 characters

fold -20 exercises.txt

13. Extracting the first n lines of big.txt

head -15 head.txt

cat head.txt|head -15

14. Seeing the last 10 lines of big.txt

tail big.txt

cat big.txt|tail

15. Show the lines of big.txt from 5th to 15th line

head -15 big.txt | tail -10

16. Generate a continuous sequence of y

yes

CTRL+c

or

yes | nl

CTRL+c

Generate a continous string of "MLRIT"

yes MLRIT

CTRL+c

17. Concatenate files in columns

paste part01 part02 part03

18. Generate numbers from 1 to 100 and keep 5 numbers per line in

fives.txt

seq 1 100 | paste - - - - - > fives.txt

19. Extract columns 1,3,5 of fives.txt

cut -f1,3,5 fives.txt

20. Extract columns from 2 to 4 of fives.txt

cut -f2-4 fives.txt

21. Generate mutilples of 2 and mutilpes of 3 in separate columns

seq -w 2 2 120 | nl > m2.txt

seq -w 3 3 120 | nl > m3.txt

join m2.txt m3.txt | more

22. Octal dump of a file

od -c fives.txt

od -a fives.txt

od -x fives.txt

od -i fives.txt

od fives.txt

23. Delete a particular chacater(s) in a file

cat fives.txt | tr -d '5'

tr -d '5' < fives.txt

Delete the chracters 0 and 5 in fives.txt and show the output

to screen

tr -d '50' < fives.txt

24. Change the all upper case letters to lower case latters in

exercises.txt

tr 'A-Z' 'a-z' < exercises.txt

25. Encrypt a file using caeser cipher.

tr 'A-Za-z' 'D-ZABCd-zabc' < exercises.txt

26. Complement the delete. Anything otherthan letters, newline and

space must be deleted in exercises.txt

tr -cd 'A-Za-z\n ' < exercises.txt

27. Squeeze out the chracter set into a single a single character.

Squeeze out all the whitespaces into a new line.

tr -s ' \t\n\r' '\n' < exercises.txt

Squeeze out any non alphabet into a new line

tr -cs 'A-Za-z' '\n' < exercises.txt

28. Take out the first and thrid fields (username, id) from /etc/passwd

file int users.txt

cut -d':' -f1,3 /etc/passwd > users.txt

Have a look at it

more users.txt

Sort the users by name

sort users.txt

Sort the users by decreasing order of userid (i.e. field2 and

numerical sort)

sort -t ":" +1 -2 -nr users.txt

29. In task 27, we have squeezed out all non alphabets into new lines

(each word per line)

tr -cs 'A-Za-z' '\n' < exercises.txt

Sort these words alphabetically

tr -cs 'A-Za-z' '\n' < exercises.txt | sort

Take out the unique words from thoses

tr -cs 'A-Za-z' '\n' < exercises.txt | sort | uniq

Count each word

tr -cs 'A-Za-z' '\n' < exercises.txt | sort | uniq -c

Find the repeated words

tr -cs 'A-Za-z' '\n' < exercises.txt | sort | uniq -d

Count the number of times duplicate words appeared(the repeated

ones)

tr -cs 'A-Za-z' '\n' < exercises.txt | sort -dc

30. Show all the distinct words (normalized to lowercase) from

exercises.txt

31. Show the common numbers from m2.txt and m3.txt (regenerate m2.txt

and m3.txt)

seq -w 2 2 120 > m2.txt

seq -w 3 3 120 > m3.txt

comm m2.txt m3.txt

comm -12 m2.txt m3.txt

32. Locate and filter according to patterns using grep

Show all the lines containing 10 in the file m2.txt

grep 10 m2.txt

cat m2.txt|grep 10

Show all the lines having the numbers that begin and end with 2

in the line from seq 1 400

seq 1 400 | egrep "^2[0-9]\*2$"

Show all the lines which do not contain the pattern 20

grep -v 20 m2.txt

Show the file names along with the line numbers which end with

50 in all part\* files

grep -n 50$ part\*

Show the line followed two consequtive lines containing the

pattern "50$" from part01

grep -A2 50$ part01

Show the one line before and one line after containing the

patter 50$

grep -B1 -A1 50$ part01

Show the lines ending with 25 or 50

egrep "25$|50$" part01

33. Echo a message to the terminal (stdout)

echo "Hello World"

Echo the user greeting with date and time (not the single

quote, it is back quote)

echo Hello $USER the time now is `date`

34. Formatted output

printf "Hello %s the time now is %s\n" $USER "`date`"

printf "%d\n" 10 20 30

printf "%x" 15

35. Look for the starting with party in the dictionary

look party

36. Show all that begin with part and at the same save a copy of it in

words-part.txt

look part | tee words-part.txt

ls word\*

more words-part.txt