|  |  |  |
| --- | --- | --- |
| **Ex: 01**  **Date:** | **Linux OS and Shell Commands** | **Reg.No:** |

**AIM :**

To Study about the Commands and the various features of the LINUX Operating System.

**INTRODUCTION :**

Linux is a Unix like operating system which was developed in 1991 by Tinnish student called LINUX TORVALDS. Since the LINUX code was freely available on the internet many programmers contributed towards its growth over the years.

**FEATURES OF LINUX :**

* Multiprogramming
* Time sharing
* Multi tasking
* Virtual memory
* Shared libraries

Some of distributions of LINUX are RED HAT, CALDERA, SUSE, SLACKWARE, etc.,

**COMPONENTS OF A LINUX SYSTEM:**

1.KERNEL:

It is responsible for all system resources by interacting directly with the hardware.

2. SHELL:

It is the user interface which provides various services to the user.

3. SYSTEM LIBARIES:

A standard set of functions through which applications interact with the Kernel.

4. SYSTEM UTILITIES:

Programs that perform individual specialized management tasks.

**THE SHELL:**

The Shell process instruction that are issued to the system by the user. The various Shell available with LINUX are

* Bourne-Shell (sh)
* Bourne-against Shell (bash)
* kern Shell (ksh)
* public domain kern Shell (pdksh)
* C Shell (csh)

The Bourne and Bash shells are the most frequently used.

The Shell can also be used as a programming language because it provides many features and special commands. Any file consisting of a sequence of commands is known as shell program or a shell script.

**VARIABLES:**

* Variables in the shell are indicated by the symbol “$” which precedes the name of the variable. A variable can either be an integer or a string and is automatically declared at its first usage.

**INPUT/OUTPUT COMMANDS:**

* Read ---- Used to read one or more variables.
* Echo ---- Print either strings or values or both.

**ARITHMETIC OPERATION:**

* Expr --- Used to evaluate the expression. Eg : x=’ expr $a + $b’;

**CONDITIONAL STATEMENTS :**

* if statement :

Syntax:

If [conditions]

Then

Commands

Else

Commands

if

2. case statement:

Syntax:

Case $variable in

Value 1) commands;

Value 2) commands;

“

“

\*) commands:

* while statement :
* Syntax:

Commands

Done

* until statement :
* Syntax:

until[conditions]

do

commands

done

* for statement:
* Syntax:

For variable in <list of values>

Do

Commands

Done

* **BREAK AND CONTINUE :**

Break - used to transfer control out of a Loop.

Continue- used to continue with the next Iteration.

* **RELATIONAL OPERATORS :**

-eq Equal to

-ne Not equal to

-gt Greater than

-ge Greater than equal to

-lt Less than

-le Less than equal to

**VI EDITOR :**

Although many Text Editors like VI, VINA, EMACS, ED and SED are available, VI Editor is the most commonly used editor in LINUX. It is full screen editor which was developed by WILLIAM JOY.

**MODELS IN VI :**

The VI Editor works in three different modes shown below :

* Input Mode :

Any key that is pressed in this Mode is entered as text. The following are some of commands.

|  |  |
| --- | --- |
| **COMMANDS** | **FUNCTIONS** |
| i | Inserts any character at Cursor position |
| a | Appends text to |
| o | Opens a New blank line |
| r | Replace a Single character |
| R | Replace more than single character |

* Ex Mode :

In this mode the commands are entered in the last line after a “:” symbol. Some of the commands used in this Mode are:

|  |  |
| --- | --- |
| **COMMANDS** | **FUNCTIONS** |
| :w | Saves file and remains in editing mode |
| :wq | Saves file and quits the editor |
| :qa | Quits editor without saving changes |

* Command Mode :

The Key used here act as commands on the text. We can involve this mode by pressing <esc> key.

|  |  |
| --- | --- |
| **COMMANDS** | **FUNCTIONS** |
| H | Moves the cursor left |
| I | Moves the cursor right |
| J | Moves the cursor down |
| K | Moves the cursor up |
| X | Deletes entire character |
| Dd | Deletes entire line |
| <ctrl-f> | Scrolls full page forward |
| <ctrl-b> | Scrolls full page backward |
| <ctrl-d> | Scrolls half page forward |
| <ctrl-u> | Scrolls half page backward |

**COMMANDS:**

**1. Calendar**

**NAME** : calendar

**(i) SYNTAX** : cal

**DESCRIPTION** : Displays a simple calendar. If arguments are not

Specified, the current month is displayed.

**EXAMPLE** : cal

**OUTPUT** :

June 2014

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

**(ii) SYNTAX** : cal year

**DESCRIPTION** : Displays calendar of that year

**EXAMPLE** : cal 2012

**OUTPUT** :

January February March

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 1 2 3

8 9 10 11 12 13 14 5 6 7 8 9 10 11 4 5 6 7 8 9 10

15 16 17 18 19 20 21 12 13 14 15 16 17 18 11 12 13 14 15 16 17

22 23 24 25 26 27 28 19 20 21 22 23 24 25 18 19 20 21 22 23 24

29 30 31 26 27 28 29 25 26 27 28 29 30 31

April May June

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 5 1 2

8 9 10 11 12 13 14 6 7 8 9 10 11 12 3 4 5 6 7 8 9

15 16 17 18 19 20 21 13 14 15 16 17 18 19 10 11 12 13 14 15 16

22 23 24 25 26 27 28 20 21 22 23 24 25 26 17 18 19 20 21 22 23

29 30 27 28 29 30 31 24 25 26 27 28 29 30

July August September

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 1

8 9 10 11 12 13 14 5 6 7 8 9 10 11 2 3 4 5 6 7 8

15 16 17 18 19 20 21 12 13 14 15 16 17 18 9 10 11 12 13 14 15

22 23 24 25 26 27 28 19 20 21 22 23 24 25 16 17 18 19 20 21 22

29 30 31 26 27 28 29 30 31 23 24 25 26 27 28 29

30

October November December

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 1 2 3 1

7 8 9 10 11 12 13 4 5 6 7 8 9 10 2 3 4 5 6 7 8

14 15 16 17 18 19 20 11 12 13 14 15 16 17 9 10 11 12 13 14 15

21 22 23 24 25 26 27 18 19 20 21 22 23 24 16 17 18 19 20 21 22

28 29 30 31 25 26 27 28 29 30 23 24 25 26 27 28 29

30 31

**(iii) SYNTAX** : cal -3

**DESCRIPTION** : Displays calendar of previous, current, next months of current

year

**OUTPUT** :

July August September

Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa

1 2 3 4 5 6 7 1 2 3 4 1

8 9 10 11 12 13 14 5 6 7 8 9 10 11 2 3 4 5 6 7 8

15 16 17 18 19 20 21 12 13 14 15 16 17 18 9 10 11 12 13 14 15

22 23 24 25 26 27 28 19 20 21 22 23 24 25 16 17 18 19 20 21 22

29 30 31 26 27 28 29 30 31 23 24 25 26 27 28 29

30

**(iv)SYNTAX** : cal month year

**DESCRIPTION** : Displays the calendar for corresponding month and year .

**EXAMPLE** : cal 4 2012

**OUTPUT** :

April 2012

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

**2. Date**

**NAME** : DATE- print or set the system date and time

**(i) SYNTAX** : date

**DESCRIPTION** : Display the current time in the given format or set the system date.

**OUTPUT** : Mon Jul 23 12:17:50 IST 2012

**(ii) SYNTAX** : date +% H

**DESCRIPTION** : Display the current hour.

**OUTPUT** : 12

**(iii) SYNTAX** : date +% h

**DESCRIPTION** : Display the current month name.

**OUTPUT** : Jul

**(iv) SYNTAX** : date +% m

**DESCRIPTION** : Display the current month number.

**OUTPUT**  : 7

**(v) SYNTAX** : date +% a

**DESCRIPTION** : Display the abbreviated weekday name.

**OUTPUT** : Mon

**(vi) SYNTAX** : date +% y

**DESCRIPTION** : Display the current year.

**OUTPUT** : 12

**(vii) SYNTAX** : date +% S

**DESCRIPTION** : Display the current second.

**OUTPUT** : 57

**3. Script**

**NAME** : SCRIPT – makes typescript of terminal session

**DESCRIPTION** : Makes a typescript of everything printed on your terminal. It is useful

for students who need a hardcopy record of an interactive session as

proof of an assignment, as the typescript file can be printed out later

with lpr(1).

**SYNTAX** : script scriptname

………..

………..

………..

Exit

**OPENING A SCRIPT:**

**SYNTAX** : vi scriptname

**EXAMPLE** : vi date.txt

**OUTPUT** : ~date

~ Mon Jul 23 12:17:50 IST 2012

~

~

INSERT

**4. ls**

**NAME** : LIST – list directory contents

**(i) SYNTAX** : ls

**DESCRIPTION** : List information about the Files (the current directory by default).

**OUTPUT** :

greatest.sh cse.txt mouse.txt digit.sh

emp.sh num.sh case.sh

**(ii) SYNTAX** : ls –l

**DESCRIPTION** : Displays files in long listing format.

**OUTPUT** :

csea08 csea08 1384962 Jul 23 12:17 [23.07.2012]

csea08 csea08 20325 Jul 23 12:30 [23.07.12]

csea08 csea08 138 Jul 23 12:33 [01;32case.sh]

csea08 csea08 13830 Jul 23 12:37 [01;34cse]

csea08 csea08 183 Jul 23 12:40 [01;32mdigit.sh]

csea08 csea08 530 Jul 23 12:50 [01;32memp.sh]

csea08 csea08 730 Jul 23 12:59 [01;32mgreatest.sh]

**(iii) SYNTAX** : ls -r

**DESCRIPTION** : Displays the files in reverse sorted order.

**OUTPUT** :

num.sh mouse.txt greatest.sh emp.sh

digit.sh cse.txt case.sh

**(iv) SYNTAX** : ls -s

**DESCRIPTION** : Displays the size of each files.

**OUTPUT** :

8 num.sh 4 mouse.txt 8 greatest.sh 8 emp.sh

8 digit.sh 4 cse.txt 8 case.sh

**(v) SYNTAX** : ls -S

**DESCRIPTION** : Displays the files in sorted order.

**OUTPUT** :

case.sh cse.txt digit.sh emp.sh

greatest.sh mouse.txt num.sh

**5. cp**

**NAME** : cp – copy files and directories

**SYNTAX** : cp fi f2

**DESCRIPTION** : Copies f1 to f2

**EXAMPLE** : cp cse.txt new.txt

**6. rm**

**NAME** : rm – remove files

**SYNTAX** : rm filename

**DESCRIPTION** : This command removes each specified file.

**EXAMPLE** : rm cse.txt

**7. mv**

**NAME** : mv – move(rename) files

**SYNTAX** : mv f1 f2

**DESCRIPTION** : Renames Source to Destination

**EXAMPLE** : mv new.txt cse.txt

**8. mkdir**

**NAME** : mkdir – makes directory

**SYNTAX** : mkdir DirectoryName

**DESCRIPTION** : Creates the directory, if they do not exist already

**EXAMPLE** : mkdir new

**9. rmdir**

**NAME** : rmdir – removes directory

**SYNTAX** : rmdir DirectoryName

**DESCRIPTION** : Removes the directory, only it is empty.

**EXAMPLE** : rm cse.txt

**10. Pwd**

**NAME** : pwd – Present Working Directory displays the name of the

current/working directory

**SYNTAX** : pwd

**DESCRIPTION** : Displays the name of the current/working directory

**OUTPUT** : \home\csea08\new

**11. Cd**

**NAME** : cd – Change Directory

**(i) SYNTAX** : cd dirname

**DESCRIPTION** : Change the directory which we use to work with.

**EXAMPLE** : cd New

**(ii) SYNTAX** : cd ..

**DESCRIPTION** : Quits from the current directory.

**(iii) SYNTAX** : cd\

**DESCRIPTION** : Returns to the home directory.

**12. Cat**

**NAME** : cat- concatenate & open files and print on the standard output

**(i) SYNTAX** : cat > filename

**DESCRIPTION** : This command is used to open a new file.

**EXAMPLE** : cat > a.txt

**OUTPUT** :

NAME : ZZZZ

ROLL NO: XX

**(ii) SYNTAX** : cat filename

**DESCRIPTION** : To view the contents of the file.

**EXAMPLE** : cat a.txt

**OUTPUT** :

NAME : ZZZZ

ROLL NO: XX

**(iii) SYNTAX** : cat f1 f2 > f3

**DESCRIPTION** : To concatenate f1 and f2 save in f3

**EXAMPLE** : cat a.txt b.txt > c.txt

**OUTPUT** :

a.txt=>

NAME : ZZZZ

ROLL NO: XX

b.txt=>

COLLEGE.SMVEC

c.txt=>

NAME : ZZZZ

ROLL NO: XX

COLLEGE.SMVEC

**(iv) SYNTAX** : cat –n filename

**DESCRIPTION** : To display the contents of the file along with the line numbers.

**EXAMPLE** : cat -n sample.txt

**OUTPUT** : 1 hai

2 how are u?

**(v) SYNTAX** : cat f1 >> f2

**DESCRIPTION** : To redirect the data from one file to another.

**EXAMPLE** : cat sample.txt new.txt

**OUTPUT** : cat new.txt

sample. txt=>

hai

how are u?

new. txt=>

hai

how are u?

**13. Whoami**

**NAME** : Displays the current user login and identity.

**SYNTAX** : whoami

**OUTPUT** : csea08

**14. Man**

**NAME** : man – help command

**SYNTAX** : man command

**DESCRIPTION** : Displays the description of a command

**EXAMPLE** : man rm

**15. Head**

**NAME** : head

**SYNTAX** : head filename

**DESCRIPTION** : Displays the first ten lines in the file.

**EXAMPLE** : head fruits.txt

**OUTPUT** : apple

banana

cherry

jack fruit

strawberry

orange

pineapple

mango

grape

papaya

**16. Tail**

**NAME** : tail

**SYNTAX** : tail filename

**DESCRIPTION** : Displays the last ten lines in the file.

**EXAMPLE** : tail fruits.txt

**OUTPUT** : apple

banana

cherry

jack fruit

strawberry

orange

pineapple

mango

grape

papaya

**17. Clear**

**NAME** : clear

**SYNTAX** : clear

**DESCRIPTION** : Clears the content of the command prompt.

**18. Sort**

**NAME** : sort

**(i) SYNTAX** : sort filename

**DESCRIPTION** : Sorts the content of the file in ascending order.

**EXAMPLE** : sort names.txt

**OUTPUT** :

Arun

Balu

Chandra

David

Edwards

**(ii) SYNTAX** : sort –r filename

**DESCRIPTION** : Sorts the content of the file in descending order.

**EXAMPLE** : sort –r names.txt

**OUTPUT** :

Edwards

David

Chandra

Balu

Arun

**19. Who**

**NAME** : who

**SYNTAX** : who

**DESCRIPTION** : Displays all the users currently logged it.

**OUTPUT** :

csea13 pts/1 Jul 23 10:08 (172.17.22.38)

csea06 pts/10 Jul 23 10:13 (172.17.21.11)

csea12 pts/12 Jul 23 10:13 (172.17.21.35)

csea18 pts/11 Jul 23 10:13 (172.17.20.9)

csea24 pts/8 Jul 23 10:14 (172.17.22.33)

csea05 pts/13 Jul 23 10:15 (172.17.21.36)

root :0 Jul 23 10:20

csea20 pts/16 Jul 23 10:22 (172.17.21.43)

csea11 pts/20 Jul 23 10:28 (172.17.21.27)

csea23 pts/5 Jul 23 10:37 (172.17.22.14)

csea03 pts/0 Jul 23 10:37 (172.17.222.37)

csea01 pts/4 Jul 23 11:00 (172.17.22.13)

csea21 pts/15 Jul 23 11:00 (172.17.21.34)

csea07 pts/18 Jul 23 11:00 (172.17.20.28)

csea08 pts/14 Jul 23 11:01 (172.17.22.16)

csea16 pts/7 Jul 23 11:01 (172.17.21.24)

csea02 pts/19 Jul 23 11:01 (172.17.21.45)

staff pts/26 Jul 23 12:20 (172.17.21.21)

**20. Finger**

**NAME** : finger

**SYNTAX** : finger

**DESCRIPTION** : Displays the detailed information about the system users.

**OUTPUT** :

Login Name Tty Idle Login Time Office Office Phone

csea01 pts/4 2 Jul 23 11:00 (172.17.22.13)

csea02 csea02 pts/19 2 Jul 23 11:01 (172.17.21.45)

csea03 pts/0 Jul 23 10:37 (172.17.222.37)

csea05 pts/13 Jul 23 10:15 (172.17.21.36)

csea06 pts/10 Jul 23 10:13 (172.17.21.11)

csea07 pts/18 Jul 23 11:00 (172.17.20.28)

csea08 pts/14 Jul 23 11:01 (172.17.22.16)

csea09 pts/3 Jul 23 12:13 (172.17.22.115)

csea10 pts/2 1 Jul 23 12:08 (172.17.22.26)

csea11 pts/20 1 Jul 23 10:28 (172.17.21.27)

csea12 pts/12 1 Jul 23 10:13 (172.17.21.35)

csea13 pts/1 1 Jul 23 10:08 (172.17.22.38)

csea14 pts/21 Jul 23 11:01 (172.17.22.29)

csea15 pts/9 Jul 23 11:25 (172.17.22.130)

root root \*:0 Jul 23 10:20

staff pts/26 1 Jul 23 12:20 (172.17.21.21)

**21. Last**

**NAME** : last

**SYNTAX** : last

**DESCRIPTION** : Displays the list of last logged-in users for a month.

**OUTPUT** :

staff pts/26 172.17.21.21 Mon Jul 23 12:20 still logged in

csea09 pts/3 172.17.22.115 Mon Jul 23 12:13 still logged in

csea04 pts/23 172.17.22.60 Mon Jul 23 12:12 - 12:18 (00:05)

csea23 pts/25 172.17.22.60 Mon Jul 23 12:11 - 12:11 (00:00)

csea10 pts/2 172.17.22.26 Mon Jul 23 12:08 still logged in

csea04 pts/23 172.17.22.60 Mon Jul 23 11:51 - 11:52 (00:00)

csea08 pts/23 172.17.22.60 Mon Jul 23 11:50 - 11:51 (00:01)

csea04 pts/23 172.17.22.60 Mon Jul 23 11:48 - 11:50 (00:01)

csea06 pts/23 172.17.22.60 Mon Jul 23 11:46 - 11:47 (00:01)

csea23 pts/6 172.17.22.14 Mon Jul 23 11:45 still logged in

csea09 pts/3 172.17.22.14 Mon Jul 23 11:44 - 11:45 (00:01)

csea09 pts/6 172.17.22.115 Mon Jul 23 11:38 - 11:42 (00:04)

reboot system boot 2.4.20-8smp Mon Jul 23 08:57 (03:24)

wtmp begins Mon Jul 2 10:08:30 2012

**22. And**

**NAME** : and - &&

**SYNTAX** : cmd1 && cmd2

**DESCRIPTION** : Used to combine more than one commands.

**EXAMPLE** : whoami && date

**OUTPUT** : csea23

Mon Jul 23 12:22:54 IST 2012

**23. Or**

**NAME** : or - ||

**SYNTAX** : cmd1 || cmd2

**DESCRIPTION** : Displays the output for one command which is true.

**EXAMPLE** : whoami || date

**OUTPUT** : csea23

**24. . Alias**

**NAME** : alias

**SYNTAX** : alias name="value"

**DESCRIPTION** : To create simple names or abbreviations for commands

**EXAMPLE** : alias p="pwd"

p

**OUTPUT** : /home/mca1

**25. Edit**

**NAME** : edit

**SYNTAX** : vi filename

**DESCRIPTION** : Edits the content of the file. To edit press I and to save press esc:wq

**EXAMPLE** : vi names.txt

**26. cut**

**NAME** : cut

**(i) SYNTAX** : cut *OPTION -m*  [*FILE*]...

**DESCRIPTION** : extracts m characters from the beginning of each line from the specified file.

**EXAMPLE** : cut –c -3 test.txt

**OUTPUT** :

**BEFORE EXECUTION** : cat > test.txt

Smvec

manakula

vinayagar

**AFTER EXECUTION** : smv

man

vin

**27. Touch**

**NAME** : touch

**SYNTAX** : touch filename

**DESCRIPTION** : Creates an empty file.

**EXAMPLE** : touch hello.txt

**OUTPUT** :

**BEFORE EXECUTION** : ls

Sample.txt welcome.txt

**AFTER EXECUTION** : ls

Sample.txt welcome.txt hello.txt

**28. Uniq**

**NAME** : uniq

**SYNTAX** : uniq option filename1 filename2

**DESCRIPTION** : Discard all but one of successive identical lines from filename1 to filename2

**EXAMPLE**  : uniq –d a.txt b.txt

**OUTPUT** :

**BEFORE EXECUTION** : cat > a.txt

smvec

smvec

manakula

vinayagar

**AFTER EXECUTION** : cat b.txt

smvec

**29. Semicolon**

**NAME** : Semicolon (;)

**SYNTAX** : cmd1 ; cmd2; cmd3

**DESCRIPTION** : Similar to ‘and’ command which combines more than one command.

**EXAMPLE** : whoami && date

**OUTPUT** : csea23

Mon Jul 23 12:22:54 IST 2012

**30. Echo**

**NAME** : echo – displays a line of text.

**SYNTAX** : echo “…….”

**DESCRIPTION** : Displays the statement within double quotes.

**EXAMPLE** : echo “hai”

**OUTPUT** : hai

**31. Word Count**

**NAME** : wc – word count

**(i) SYNTAX** : wc filename

**DESCRIPTION** : Displays the number of lines, words and characters in files.

**EXAMPLE** : wc a.txt

**OUTPUT** : a.txt=>

hai

how are u

2 lines, 4 words, 12 characters.

**(ii) SYNTAX** : wc -l filename

**DESCRIPTION** : Displays the number of lines in files.

**EXAMPLE** : wc –l a.txt

**OUTPUT** : a.txt=>

hai

how are u

2 lines

**(iii) SYNTAX** : wc -m filename

**DESCRIPTION** : Displays the number of characters in files.

**EXAMPLE** : wc –m a.txt

**OUTPUT** : a.txt=>

hai

how are u

12 characters

**(iv) SYNTAX** : wc -w filename

**DESCRIPTION** : Displays the number of words in files.

**EXAMPLE** : wc –w a.txt

**OUTPUT** : a.txt=>

hai

how are u

4 words

**32. Grep**

**NAME** : grep

**(i) SYNTAX** : grep pattern filename

**DESCRIPTION** : To search for a regular expression or a pattern in a file

**EXAMPLE** : grep apple b.txt

**OUTPUT** : b.txt=>

This is an apple

This is a fruit

apple

**(ii) SYNTAX** : grep –c pattern filename

**DESCRIPTION** : To search for a regular expression or a pattern in a file and displays

how many times that pattern is repeated in the file.

**EXAMPLE** : grep –c apple b.txt

**OUTPUT** : b.txt=>

This is an apple

This is a fruit

1

**(iii) SYNTAX** : grep –n pattern filename

**DESCRIPTION** : To search for a regular expression or a pattern in a file and displays

the searched content along with the line and line number, if found.

**EXAMPLE** : grep –n apple b.txt

**OUTPUT** : b.txt=>

This is an apple

This is a fruit

1: apple

**(iv) SYNTAX** : grep –i pattern filename

**DESCRIPTION** : To search for a regular expression or a pattern in a file irrespective of

the case.

**EXAMPLE** : grep –n APPLE a.txt

**OUTPUT** : b.txt=>

This is an apple

This is a fruit

apple

**33. Read**

**NAME** : read – reads a value(s)

**SYNTAX** : read identifier

**DESCRIPTION** : Reads a value(s)

**EXAMPLE** : read a

**OUTPUT** : 10

read a

**34. Fgrep**

**NAME** : fgrep

**(i)SYNTAX** : fgrep pattern f1 f2

**DESCRIPTION** : To search for a regular expression or a pattern in two files

**EXAMPLE** : fgrep hai a.txt d.txt

**OUTPUT** : a.txt=>

hai

how are u

d.txt=>

this is a file

the type of the file is text

a.txt: hai

**(ii) SYNTAX** : fgrep –c pattern f1 f2

**DESCRIPTION** : To search for a regular expression or a pattern in two files and

displays how many times that pattern is repeated in the files.

**EXAMPLE** : fgrep –c file a.txt d.txt

**OUTPUT** : a.txt=>

hai

how are u

d.txt=>

this is a file

the type of the file is text

a.txt : 0

d.txt : 2

**(iii) SYNTAX** : fgrep –n pattern f1 f2

**DESCRIPTION** : To search for a regular expression or a pattern in two files and

displays the searched content along with the line and line number, if

found.

**EXAMPLE** : fgrep –c file a.txt d.txt

**OUTPUT** : a.txt=>

hai

how are u

d.txt=>

this is a file

the type of the file is text

d.txt : 1 : this a file

d.txt : 2 : the type of the file is text

**(iv) SYNTAX** : fgrep –i pattern f1 f2

**DESCRIPTION** : To search for a regular expression or a pattern in two files

irrespective of the case.

**EXAMPLE** : fgrep –c HaI a.txt d.txt

**OUTPUT** : a.txt=>

hai

how are u

d.txt=>

this is a file

the type of the file is text

a.txt : hai

**35. Pipe**

**NAME** : Pipe - |

**SYNTAX** : cmd1 | cmd2 | cmd3

**DESCRIPTION** : Makes the output of one command as input for another command.

**EXAMPLE** : date | wc -w

**OUTPUT** : 6

**36. Tee**

**NAME** : Tee

**SYNTAX** : cmd1 | tee filename

**DESCRIPTION** : Used to read the standard input and then write to standard output or

file.

**EXAMPLE** : date | tee f.txt | wc -w

**OUTPUT** : 6

**37. Write**

**NAME** : write

**SYNTAX** : write login\_name

**DESCRIPTION** : Used to communicate with other logged in users.

**EXAMPLE** : write csea25

**38. Mail**

**NAME** : Mail

**(i) SYNTAX** : mail login\_name

**DESCRIPTION** : Used to send mail to a user.

**EXAMPLE** : write csea25

**(ii) SYNTAX** : mail

**DESCRIPTION** : Used to view the mails in the mailbox.

**39. Terminal Name**

**NAME** : tty

**SYNTAX** : tty

**DESCRIPTION** : Used to display the terminal path name.

**OUTPUT** : /dev/pts/23

**40. Expression**

**NAME** : expr

**SYNTAX** : `expr expression`

**DESCRIPTION** : Used to evaluate an expression

**EXAMPLE** : echo `expr 10 + 10`

**OUTPUT** : 20

**41. . free**

**NAME** : free

**SYNTAX** : free

**DESCRIPTION** : Displays the total amount of free and used physical and swap memory in the system

**EXAMPLE** : free

**OUTPUT** :

[mca1@redhat ~]$ free

total used free shared buffers cached

Mem: 1016820 647688 369132 0 59916 414316

-/+ buffers/cache: 173456 843364

Swap: 2064380 0 2064380

**42. Find**

**ping**

**NAME** : ping

**(i) SYNTAX** : ping Ip-address

**DESCRIPTION** : Ping is used diagnostically to ensure that a [host](http://searchcio-midmarket.techtarget.com/definition/host) computer the user is trying to reach is actually operating.

**EXAMPLE** : ping 172.17.10.111

**OUTPUT** :

[mca1@redhat ~]$ ping 172.17.10.111

PING 172.17.10.111 (172.17.10.111) 56(84) bytes of data.

64 bytes from 172.17.10.111: icmp\_req=1 ttl=64 time=0.032 ms

64 bytes from 172.17.10.111: icmp\_req=2 ttl=64 time=0.017 ms

64 bytes from 172.17.10.111: icmp\_req=3 ttl=64 time=0.013 ms

^Z

[1]+ Stopped ping 172.17.10.111

**43. Password**

**NAME** : Password

**SYNTAX** : passwd

**DESCRIPTION** : Used to change the password.

**EXAMPLE** : passwd

**OUTPUT** : Changing password for user csea09. Changing password for csea09 (current) UNIX password: New password:

**44.Exit**

**NAME** : Logout

**SYNTAX** : logout

**DESCRIPTION** : Used to logout.

**45.Uname**

**NAME** : Uname

**SYNTAX** : uname

**DESCRIPTION** : Used to display the name of the system being used.

**OUTPUT** : Linux

**46. Sequence**

**NAME** : seq

**SYNTAX** : seq starting\_value ending\_value

**DESCRIPTION** : print a sequence of numbers

**EXAMPLE** : seq 1 4

**OUTPUT** :

1

2

3

4

**47. Open and Fill**

**NAME** : Open and Fill

**SYNTAX** : vi +filename

**DESCRIPTION** : Used to open a file and position the cursor at the first line.

**EXAMPLE** : vi +b.txt

**OUTPUT** :

hi bala

:wq "new.txt" 7L, 42C written

**48. Compare**

**NAME** : Compare

**SYNTAX** : comm filename1 filename2

**DESCRIPTION** : Compares the sorted files line by line.

**EXAMPLE** : comm a.txt b.txt

**OUTPUT** : OS is a program that acts as an interface.

**49. Type**

**NAME** : Type

**SYNTAX** : type filename

**DESCRIPTION** : To view the content of the file.

**EXAMPLE** : type names.txt

**OUTPUT** :

Arun

Balu

Chandra

David

Edwards

**50. Basic Calculator**

**NAME** : Basic Calculator

**SYNTAX** : bc

**DESCRIPTION** : To open the basic calculator.

**EXAMPLE** : bc

**OUTPUT** : 10 + 10

20

Result: