Few Commonly used shell commands and utilitities

1. test command : This is a very useful shell program and is often used in the conditional part of the if-statement constructs. The test command is used to check file types and compare values.
Syntax of use : **test** EXPR **Exit status:** 0 if the expression is true, 1 if the expression is false, 2 for error
test has **file status** checks, **string operators**, and **numeric comparison operators**.

EXPR	File status Semantics
-d FILE	True if FILE exists and is a directory
-f FILE	True if FILE exists and is a regular file
-L FILE	True if FILE exists and is a symbolic link

String tests : These options test string characteristics.

EXPR (shell)	Semantics
-z STRING	length (STRING) is equal to zero
-n STRING	length (STRING) is not equal to zero
STRING1 == STRING2	True if the strings are equal (synonym for =)
STRING1 != STRING2	True if the strings are not equal

Numeric tests : Numeric relational operators. The arguments must be entirely numeric (possibly negative).

EXPR	Numeric tests
ARG1 -eq ARG2	ARG1 == ARG2
ARG1 -ne ARG2	ARG1 != ARG2
ARG1 -lt ARG2	ARG1 < ARG2
ARG1 -le ARG2	ARG1 ≤ ARG2
ARG1 -gt ARG2	ARG1 > ARG2
ARG1 -ge ARG2	ARG1 ≥ ARG2

File characteristic tests : These options test other file characteristics.

EXPR	Semantics
-e FILE	True if FILE exists
-s FILE	True if FILE exists and size > zero

EXPR	File Access Permission Semantics
-r FILE	True if FILE exists and read permission is granted
-w FILE	True if FILE exists and write permission is granted
-x FILE	True if FILE exists and execute permission is granted (or search permission if it is a directory)

Connectives for test : The usual logical connectives as given below.

EXPR	Connectives (Semantics)
! EXPR	True if EXPR is false
EXPR1 -a EXPR2	True if both EXPR1 **and** EXPR2 are true
EXPR1 -o EXPR2	True if either EXPR1 **or** EXPR2 is true

2. read command- Read a line from standard input; reads the value of a variable from stdin, that is, interactively fetches input from the keyboard.
Syntax: read [options] var**Option Meaning**
-d DELIM The first character of DELIM is used to terminate the input line, rather than newline.
-s Silent mode. If input is coming from a terminal, characters are not echoed.
-t TIMEOUT Cause read to time out and return failure if a complete line of input is not read within TIMEOUT seconds. This option has no effect if read is not reading input from the terminal or from a pipe.

3. cut command : Select a Specific Field from a File
Syntax: cut [options] file
Option -f specifies the field we want to extractoption -d specifies what is the field delimiter that is used in the input file.
Examples : To set delimiter for cut to ':' use -d':' for using space, use -d' '
-f1 chooses the first field; -f1-4,6,7 denotes multiple field selection – fields 1, 2, 3 ,4, 6 and 7
4. paste command : Merge lines from files
Syntax : paste [options] files option -d specifies the delimiter to be used for pasting fields
Example : Let the contents of a file input be :

```
1 2 3
a b c
# $ %
```

each character is seperated by one space

Execute the two commands

```
cat input |  cut -d' ' -f1 > h1
cat  input | cut -d' ' -f3 > h2
```

Contents of h1

```
1
a
#
```

Contents of h2

3 c Execute the command paste -d' ' h2 h1	% Display on screen 3 1 c a % #

The single space separating the fields of h2 and h1 is because -d' '; if we wished to have 3 spaces to separate them, use -d '   '

5. sort command : sort lines of a file

syntax : sort [options] file

The sort command sorts its input in ascending order

options : -n for numeric sort   -kc for column number c

-r for reverse sort (descending order)

-k24 (muliple column - first on col2 then on col 4)

-u unique, i.e., sorted output without duplicates

-M : Sorts based on months, only first 3 letters as month, JAN, FEB

-n : Uses numeric sorting

-r : Reverse order sorting

-k : Sorts file based on the data in the specified field positions.

6. sed : Stream-line Editor for filtering and transforming text ; Reads line by line, possible to edit a line as specified

syntax : sed options file

options : s /regexp/replacement/g   substitute (s) all instances (g for global) of the pattern **regexp** in the line to be replaced by the **replacement**

7. expr command : evaluate expressions

syntax : expr EXPR

EXPR	**Semantics**		**EXPR**	**Semantics**
ARG1 < ARG2	ARG1 is less than ARG2		ARG1 + ARG2	arithmetic sum of ARG1 and ARG2
ARG1 <= ARG2	ARG1 is less than or equal to ARG2		ARG1 - ARG2	arithmetic difference of ARG1 and ARG2
ARG1 = ARG2	ARG1 is equal to ARG2		ARG1 * ARG2	arithmetic product of ARG1 and ARG2
ARG1 != ARG2	ARG1 is unequal to ARG2		ARG1 / ARG2	arithmetic quotient of ARG1 divided by ARG2
ARG1 >= ARG2	ARG1 is greater than or equal to ARG2		ARG1 % ARG2	arithmetic remainder of ARG1 divided by ARG2
ARG1 > ARG2	ARG1 is greater than ARG2			

8. grep command   : print lines matching a pattern

syntax : grep [OPTIONS] PATTERN [FILE...]   grep scans the lines in its input for the pattern and displays those lines that match

PATTERN : The caret ^ and the dollar sign $ are meta-characters that respectively match the empty string at the beginning and end of a line.

Common use : ls -l | grep '^d'   displays all directory entries because such entries have a 'd' at start of line

ls -l | grep '.dot$' displays all filenames in the directory ending with .dot

9. Other Commonly used commands :

Command	**Usage (limited description) and semantics**
cat	cat [options] filename : concatenates the contents of the file to the standard output
date	date : displays date in the format : day Mon date time zone year : Fri Nov 22 21:35:36 IST 2013
rm	rm [options] comma separated files : deletes all the file in the argument from the file system
head	head [option] file : displays the first few lines (option -n10 displays first 10 lines) to stdout
tail	tail [option] file : displays the last few lines (option -n10 displays last 10 lines) to stdout
uniq	uniq [option] file : matching adjacent lines are merged to first occurence; omits repeated adjacent lines
tr	tr [option] set1 set2 ; translate each character in set1 by the corresponding in set2; tr A-Z a-z file : converts each uppercase character in file to lowercase