## Exp. No. 1 Basic Unix Commands

**Date :**

### Aim

To study and execute Unix commands.

### Login

Type **telnet** *server_ipaddress* in **run** window.

User has to authenticate himself by providing *username* and *password*. Once verified, a greeting and **$** prompt appears. The shell is now ready to receive commands from the user. Options suffixed with a hyphen (–) and arguments are separated by space.

### General commands

| **Command** | **Function** |
|---|---|
| `Date` | Used to display the current system date and time. |
| `date +%D` | Displays date only |
| `date +%T` | Displays time only |
| `date +% Y` | Displays the year part of date |
| `date +% H` | Displays the hour part of time |
| `Cal` | Calendar of the current month |
| `cal` *year* | Displays calendar for all months of the specified year |
| `cal` *month year* | Displays calendar for the specified month of the year |
| `Who` | Login details of all users such as their IP, Terminal No, User name, |
| `who am i` | Used to display the login details of the user |
| `Uname` | Displays the Operating System |
| `uname -r` | Shows version number of the OS (kernel). |
| `uname -n` | Displays domain name of the server |
| `echo $HOME` | Displays the user's home directory |
| `Bc` | Basic calculator. Press `Ctrl+d` to quit |
| `lp` *file* | Allows the user to spool a job along with others in a print queue. |
| `man` *cmdname* | Manual for the given command. Press `q` to exit |
| `history` | To display the commands used by the user since log on. |
| `exit` | Exit from a process. If shell is the only process then logs out |

### Directory commands

| **Command** | **Function** |
|---|---|
| `Pwd` | Path of the present working directory |
| `mkdir` *dir* | A directory is created in the given name under the current directory |
| `mkdir` *dir1 dir2* | A number of sub-directories can be created under one stroke |
| `cd` *subdir* | Change Directory. If the *subdir* starts with / then path starts from **root** (absolute) otherwise from current working directory. |
| `cd` | To switch to the home directory. |
| `cd /` | To switch to the root directory. |
| `cd ..` | To move back to the parent directory |
| `rmdir` *subdir* | Removes an empty sub-directory. |

### File commands

| Command | Function |
|---|---|
| `cat` > *filename* | To create a file with some contents. To end typing press `Ctrl+d`. The > symbol means redirecting output to a file. (< for input) |
| `cat` *filename* | Displays the file contents. |
| `cat` >> *filename* | Used to append contents to a file |
| `cp` *src des* | Copy files to given location. If already exists, it will be overwritten |
| `cp -i` *src des* | Warns the user prior to overwriting the destination file |
| `cp -r` *src des* | Copies the entire directory, all its sub-directories and files. |
| `mv` *old new* | To rename an existing file or directory. `-i` option can also be used |
| `mv` *f1 f2 f3 dir* | To move a group of files to a directory. |
| `mv -v` *old new* | Display name of each file as it is moved. |
| `rm` *file* | Used to delete a file or group of files. `-i` option can also be used |
| `rm *` | To delete all the files in the directory. |
| `rm -r *` | Deletes all files and sub-directories |
| `rm -f *` | To forcibly remove even write-protected files |
| `Ls` | Lists all files and subdirectories (blue colored) in sorted manner. |
| `ls` *name* | To check whether a file or directory exists. |
| `ls` *name** | Short-hand notation to list out filenames of a specific pattern. |
| `ls -a` | Lists all files including hidden files (files beginning with **.**) |
| `ls -x` *dirname* | To have specific listing of a directory. |
| `ls -R` | Recursive listing of all files in the subdirectories |
| `ls -l` | Long listing showing file access rights (read/write/execute-**rwx** for user/group/others-**ugo**). |
| `cmp` *file1 file2* | Used to compare two files. Displays nothing if files are identical. |
| `wc` *file* | It produces a statistics of lines (**l**), words(**w**), and characters(**c**). |
| `chmod` *perm file* | Changes permission for the specified file. (`r=4, w=2, x=1`)<br>`chmod 740` *`file`* sets all rights for user, read only for groups and no rights for others |

The commands can be combined using the pipeline (`|`) operator. For example, number of users logged in can be obtained as.

```
who | wc -l
```

Finally to terminate the unix session execute the command **exit** or **logout**.

**Output**

```
$ date
Sat Apr  9 13:03:47 IST 2011

$ date +%D
04/09/11

$ date +%T
13:05:33
```

```
$ date +%Y
2011

$ date +%H
13


$ cal 08 1998
     August 1998
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

$ who
root     :0           Apr  9 08:41
vijai    pts/0        Apr  9 13:00 (scl-64)
cse4001  pts/3        Apr  9 13:18 (scl-41.smkfomra.com)

$ uname
Linux

$ uname -r
2.4.20-8smp

$ uname -n
localhost.localdomain


$ echo $HOME
/home/vijai

$ echo $USER
vijai

$ bc
3+5
8


$ pwd
/home/vijai/shellscripts/loops

$ mkdir filter
$ ls
filter  list.sh  regexpr  shellscripts

$ cd shellscripts/loops/
```

```
$

$ cd
$

$ cd /
[vijai@localhost /]$

[vijai@localhost /]$ cd /home/vijai/shellscripts/loops/
$ cd ..
[vijai@localhost shellscripts]$

$ rmdir filter
$ ls
list.sh  regexpr  shellscripts

$ cat > greet
hi cse
wishing u the best

$ cat greet
hi ece-a
wishing u the best

$ cat >> greet
bye
$ cat greet
hi cse
wishing u the best
bye

$ ls
greet  list.sh  regexpr  shellscripts

$ ls -a
.              .bash_logout   .canna  .gtkrc   regexpr       .viminfo.tmp
..             .bash_profile  .emacs  .kde     shellscripts  .xemacs
.bash_history  .bashrc        greet   list.sh  .viminfo


$ ls -l
-rw-rw-r--    1 vijai    vijai          32 Apr 11 14:52 greet
-rw-rw-r--    1 vijai    vijai          30 Apr  4 13:58 list.sh
drwxrwxr-x    2 vijai    vijai        4096 Apr  9 14:30 regexpr

$ cp greet ./regexpr/
$ ls
greet  list.sh  regexpr  shellscripts
$ ls ./regexpr
```

```
demo greet

$ cp -i greet ./regexpr/
cp: overwrite 'greet'? n

$ mv greet greet.txt
$ ls
greet.txt  list.sh  regexpr  shellscripts

$ mv greet.txt ./regexpr/
$ ls
list.sh  regexpr  shellscripts

$ rm -i *.sh
rm: remove regular file 'fact.sh'? y
rm: remove regular file 'prime.sh'? y
$ ls
list.sh  regexpr  shellscripts

$ wc list.sh
      4       9      30 list.sh
$ wc -l list.sh
      4 list.sh

$ cmp list.sh fact.sh
list.sh fact.sh differ: byte 1, line 1

$ ls -l list.sh
-rw-rw-r--    1 vijai    vijai          30 Apr  4 13:58 list.sh

$ chmod ug+x list.sh

$ ls -l list.sh
-rwxrwxr--    1 vijai    vijai          30 Apr  4 13:58 list.sh
$ chmod 740 list.sh
$ ls -l list.sh
-rwxr-----    1 vijai    vijai          30 Apr  4 13:58 list.sh
```

**Result**

Thus the study and execution of Unix commands has been completed successfully.