Lecture 10: Beginning with UNIX (login/logout)

Login and Logout Procedure

- User can log in from any terminal of an initialized system
- The login: prompt appears for the user to enter his / her user-name
- User-name and the password must be entered correctly
- Every response must be terminated by pressing the Return key
- After successful login, the Operating Systems prompt will appear on the screen
- The work session continues until the user logs out
- To Logout from a UNIX system, press CTRL+D (Control and D) keys.

The **UNIX** system restricts system access only to authorized users of the system, i.e., only those users, who have valid accounts, are allowed to work on the UNIX system. A special user, who is responsible for administrating the UNIX system, called the Superuser or System Administrator, creates these login accounts.

Login

The moment the terminal is switched on, the login prompt is displayed where the user can type his/her user-name and press <Enter>.

After the UNIX system reads the user-name,)t usually prompts for a password as shown below.

login: student
password:

Password

Associating passwords along with the login names, allow a user and the system an addition91 measure of security. However, having a password may not be mandatory on a particular system.

Password, if needed, is not echoed on the terminal while typing. Combination of login name and password is used to check the user's authorization.

If both the entries are entered correctly, the Shell prompt appears, otherwise the user is asked to re-enter the user name followed by the password. The prompt is system and user specific. In most systems it is a $ for the general user.

It is important to remember that the user-name (also called user-id) and password must be spelt exactly as recorded in the computer. For UNIX upper and lower are always distinct and different for the purpose of checking spellings.

Logout

To end a session the user has to Logout from the system.

This can be done by pressing the **Control** and D keys **(CTRL+ D)** simultaneously.

Introductory Commands

- date
- cal
- banner
- write
- who
- mesg
- passwd

Note: In all **UNIX** commands, upper and lower case letters are different. Typing **DATE** will evoke the response "DATE not found".

.**date**

Display the system date and time

Syntax - $ date [+format] ($ is the system prompt)

Format directives of date

%D : displays date as mm/dd/yy

%a : displays abbreviated weekdays (Sun to Sat)

%h : displays abbreviated month (Jan to Dec)

%m : displays the month of year (01 to 12)

%d : displays the day of the month (01 to 31)

%y : displays last two digits of the year (00 to 99)

Format directives of Time

%T : displays the time as HH:MM:SS

%H : displays the hour as (00 to 23)

%M : displays the minute (00 to 59)

%S : displays the seconds (00 to 59)

Internally the date and time is maintained by the real time clock present in the CPU of any computer. The Operating System makes use of it for various purposes.

UNIX uses the date and time to schedule different processes; keeping track of the time taken by each process and for login accounting etc.

To display the system date and time, the date command is used. Note that any user can display the date and time but only the Super user can modify it.

The date command has different format directives, using them the output can be changed according to the need.

The Format directives for date is used to display the output in specified formats: Different formats begin with a plus (+) sign. These format directives can be combined with one another.

Examples:

$ date <Enter>
Mon Mar 27 12:44:04 IST 1995 (this is the default format)

date only shows the Day, Month of the year, Date, Time, Time standard and the Year

$ date +%D	<Enter>
06/29/95	The output will be in mm/dd/yy format

$ date +%T	<Enter>
11:56:34	Will display the current time

```
$ date +'%d %D %T'                    <Enter>
29 06/29/95 12:01:11
```

More than one format directive can be combined by enclosing them within quotes. Output will be the combination of output from different formats.

cal

This command is used to display the calendar for a month or for the entire year.

```
$ cal [[mm][yyyy]]
where

mm stands for month number from 1 to 12
yyyy stands for year from 1 to 9999
```


The year and the month specifications must be numeric.

The various formats possible with the cal command are as follows:

```
$ cal                          will display calendar for the current or three months
$ cal <year>                   will display calendar for specified year
$ cal <month> <year>           will display calendar for specified month and year.
```

Example:

```
$ call 1985                    <Enter>
```

The above command will display the calendar for January 1985

January 1985

S	M	Tu	W	'l'h	F	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		

Question:
Write the command to display the calendar for September 1911.

banner

This command is useful for displaying welcome messages on the terminal. It explodes _ arguments into larger size and writes.

It displays one word per line unless we enclose the arguments in either single or double quotes.

$ banner welcome to RU <Enter>

If the arguments entered are long, the banner command splits the output on the individual WC": boundaries. This makes the screen display clearer.

$ banner "welcome to RU" <Enter>

If the same argument is issued within quotes the output will be displayed on the same line

instead of being displayed on different lines. However, if the length of the string exceeds ten characters then only first ten characters will be displayed.

write

The write command permits direct communication between two terminals that are current logged on to the same UNIX System.

In write, all lines of text that are typed at the users keyboard will be sent to the recipient's terminal screen. The messages will appear on the recipient's screen on a line-by-line basis and the session is terminated on pressing Control + D key at the sender's keyboard.

The command line format is:

$ write <login name of the addressee> <Enter>

Example:

To write to a user called raj, the following command can be given:

$ write raj <Enter>
Hi! I am going to have lunch <Enter>
Are you coming along?? <Enter>
<ctrl d>
$

If the user we are sending the message to, is logged in on more than one terminal, we need to specify the name of the terminal along with the login name.

mesg

Some times the user wants to disable his/her terminal so that messages from other users do not disturb him/her while executing an important program or editing a text through a text editor. This can be achieved by using the **mesg** command.

$ mesg n will disable the terminal for write.

$ mesg y will make the terminal write enable.

Invoking **mesg** without an argument will display the status of the terminal's write protection:

y indicates write enabled.

n indicates write protected or disabled.

Question: Write a command to send messages to a user "mano", who had logged into two different terminals.