**UNIX**

Trainer : Mr.**Selvakumar Masilamani**  *Date : 15-JUN-2015*

*Time :10.30- 12.45 a.m. Day : Monday*

**INTRODUCTION :**

* Operating system are the building blocks of computer systems, & they provide the interface between the user application & computer hardware.Solaris is a multiuser-multi tasking operating system developed and sold to SUN Microsystems
* Operating System is a set of programs 🡪interface between User & Hardware
* POS 🡪 Power On System
* Files will be loaded from secondary memory to primary memory & started getting executed.
* A program in execution is called PROCESS.
* Username & password 🡪Authentication 🡪Loads user profile🡪User interface (desktop)

**SERVICES OF OPERATING SYSTEM :**

* 1)File Manager. [ Unique Identification Number 🡪 inode Number is given for the newly created file

inode Table 🡪 Array of Structure]

* 2)Process.[Proctab contains all the process table]
* Proctab contains 🡪 pid,ppiel{parent Process id},gid,uid
* 3)Memory.
* 4)Scheduling.
* 5)Inter PC.

**OPERATING SYSTEM TYPES :**

* Single user Single Task🡪 DOS
* Single user Multi task🡪 windows xp, window 7, windows8
* Multi user Multi task 🡪Unix
* AIX 🡪 Product of IBM

System Call🡪 It is a Kernel(system) level function.

Command 🡪 Program which executes a particular task.

Shell script 🡪 series or sequence of command.

We can write java ,c , shell script in UNIX

*Time : 2.35-3.45pm*

**FILE DESCRIPTOR**:

fp = fopen(“f1.txt………r”)

File pointer 🡪 index[begins with 0] to point a file which is accessed throughout the program.

File Descriptor 🡪 small index integral or integer value which is assigned to a open file.

Fd =fopen (“f1.txt”, o\_RDONLY)

**Default file open 0🡪 stdin,1🡪stdout,2🡪stderr file will occupy least position**

**REDIRECTION :**

**TYPES:**

* Input redirection
* Output direction
* Instead of getting a input through keyboard we are getting a file
* Instead of displaying a output through screen we are getting a file

**SERIALISATION :**

Storing objects into a stream as a secondary storage.

Signal 🡪 to terminate the process by communication.

**Trainer : Mr.Selvakumar Masilamani**

Date : 16-JUN-2015 Time : 10.10-11.30

**TYPES OF APPLICATIONS :**

1. Console application🡪no graphical interface [faster].
2. Windows or Desktop application 🡪 very good graphical interface [Standalone application so requires software installation].
3. Web Application 🡪installed in server & it connects device(people) & access it from anywhere.

**ADVANTAGES OF UNIX :**

* **Openess** , source code is publically available.
* **Utilities**, variety of of commercial software available
* **Portability**, every kind of supports unix
* **Multiuser**
* **Multitasking**
* **Networking**🡪unix protocol forms the base for the internet
* **Prevalence**🡪solve many complicated engg problems

**USES OF UNIX :**

1. User support tools🡪 text processing[vi,sed,awk], Productivity applications
2. Programmer Support tools🡪 Programming Language & compilerd [C,C++,Java]
3. Unix as server🡪web server,mail server, application server.

**IDE** 🡪 Integral Developing Environment [tools for writing or compiling an application]

**VI**🡪 similar to notepad

**SED**🡪 text editor

**TYPES OF COMPILER :**

1. cc
2. gcc
3. g++

**Shell script** 🡪 sequence of command,used to automate various task..

**Shell** 🡪 interface between user and kernel[**CLI**-**command line intrepreter**], to perform various task.

**KERNEL :**

It is the hub of OS. It allocates time & memory to programs & handles the filestore & communications in response to system calls.

The shell and Kernel work together.

* SLA🡪Service Level Agreement.

The UNIX operating system is made up of three parts;

* the kernel,
* the shell and
* the programs.

Time :4.00- 5.45pm

**DIFFERENCE BETWEEN LINUX & UNIX:**

* Both are open source
* Unix has firewall by default.

**TYPES OF SHELL:**

1. Bourne shell[sh]🡪original unix shell[limitation]
2. C shell[csh]🡪coomonly used as login shell.
3. Korn shell[ksh]🡪Added features,supports all sh & many C shell.
4. BASH shell{Bourne Again sh}🡪 Default shell.

Namespace 🡪 to group up the related new class.

System.out.println🡪system is package, out is class, println is method

Mount point🡪locate the File system is location.

Path name🡪absolute & relative.

* Absolute path begins with the parent of all the directories
* Relative path is with the current working directory

**Trainer : Mr.Selvakumar Masilamani**

Date : 17-JUN-2015 Time : 10.55-11.50am

|  |  |  |
| --- | --- | --- |
| S.NO | Directory | Description |
| 1) | Root | Contains OS directory, default directory |
| 2) | Export | Files to be exported while using the services like NFS,NIS, Samba |
| 3) | Export/home | Contains solaris user’s home directory |
| 4) | Lib | Library function for storage of module as well as holds kernel info |
| 5) | Dev | Logical naming convention ,controls various devices like HDD, printer. |
| 6) | Device | Contains physical naming convention |
| 7) | Usr | Contains sytem configuration information & commands |

Other directories are🡪 usr/bin, usr/sbin, tmp, etc, var , mnt ,opt ,proc

* /etc directory 🡪contains all system & services configuration etc/passws, etc/shadow, etc/inittab, etc/vfstab, etc/dfstab.
* /etc/profile🡪 contains all default profile details.
* Lib🡪 contains physical name.
* Dev🡪contains logical name.
* Usr🡪contains bin & sbin.
* Relationship between dev & device is called **LINKING.**
* Symbolic link or soft link 🡪contains the file path[eg: file shortcut]
* Source🡪content[inode number]eg{12345}
* Destination🡪Address[diff inode number]eg{12644}
* Hard link🡪
* Source🡪 content[inode number]eg{12345}
* Destination🡪 Address[same inode number]eg{12345}

Time : 12.10-11.50pm

**COMMANDS:**

Command is an program which accepts zero or more arguments.

**Syntax🡪 Command [option] [arguements]**

Options🡪are used to execute particular code to get the customized results.

Grep 🡪 command used to found a pattern, if its found then displayed as a line.

* Pwd🡪print[present] working directory, it returns an absolute path.
* mkdir🡪to create a new directory.
* cd🡪change directory.
* . 🡪current directory
* .. 🡪 previous directory.
* ~ 🡪 home directory.
* If the file is existing , it will only modify only the date.
* Ls🡪list all the files &directory.
* Ls-a 🡪 list all the files &directory including hidden files
  + Hidden files will begins with the **dot.**
* Ls-F 🡪 used to identify the file time.
  + If a file name ends with a slash , then it’s a directory
  + If its end with asterisk # , then it is a executable file
  + If it ends with @ , then it is symbolic files
* Ls-I 🡪 display the inode number of a file.
* Ls-S 🡪 displays the file size.
* Ls-r 🡪 display files in reverse order
* Ls-R 🡪 it displays the regersive directory
  + Regresive🡪files are available in current as well as sub directory
* Ls-L 🡪stands for long list, gives more info about the file.
  + If a line begins with d🡪 directory,
  + If a line begins with L-🡪 symbolic file
  + If a line begins with - 🡪normal file
* Example🡺 rw-r- -r- - [rw- 🡺 owner, r- - 🡪 group, r-- 🡪 other group]
  + R🡪 read
  + W🡪write
  + X🡪execute
* Chown🡺 Only admin can change the ownership of the file
* Chgrp🡺 Only admin can change the group.
* Umash🡪command used to assign a default permission for a file.
* Chmode🡪 command to change the permissionfo a file.

**TYPES OF VARIABLE** :

* Environment variable
* The values of environment variables are inherited by login shell
* Local variable

**SYMBOLIC MODE COMMAND SYNTAX:**

* u🡪owner
* g🡪group
* o🡪other
* a🡪all
* +🡪add
* - 🡪remove
* = 🡪absolutely overwrite the file.

|  |
| --- |
| * Host name =S175n04 * Username = inatrn61 [42] * Password = welcome1 |

Trainer : Mr.Selvakumar

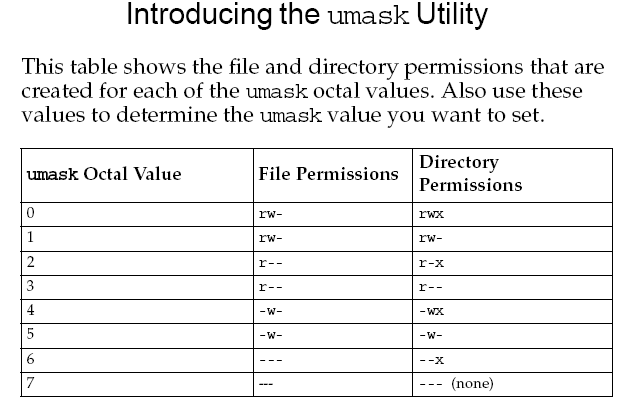
Date : 18/06/2015 Time : 9.50-10.45a.m

It deals with the octal number so it ranges from [0-7]

* 4🡪 Read
* 2🡪write
* 1🡪execute
* Read & write🡪 4+2=6
* Read , wite & execute 🡪4+2+1=7

**UMASK :** 🡪 user mask

* set system-wide default permission for all new files created by a specific user.



1. **Cat**🡪 used to disply the content of the file
   1. To connect one or more files in a series [merge]
   2. Cat-n filename 🡪 display the line number
2. **More**🡪 Used to display the file content by reading the file **content line by line.**
3. **Less**🡪display the file content & also changes the file content
4. **Pg**🡪Used to read the file content pagewise
5. **Head**🡪Displays first ten lines by default
6. **Tail**🡪Displays last ten lines by default

Example:

* Tail +51 file.txt | head -10 🡺Head -60 file.txt | tail -10 🡺 Head -60 file.txt | tail +51
* The above expressions display the same result of line from 51- 60
* Rm –i 🡪 to remove the file with **confirmation** of deleting a file.
* Rm –f 🡪 it forces the protected file to delete.
* Rm dir🡪 used to delete only an empty directory.
* Rm-r 🡪 used to delete the normal[non empty] file.
* Cp 🡪 to copy the file.
* Example : cp merge.sh newmerge.sh
* Cp-r 🡪 to copy the directory.

**Mv** 🡪 used to move or rename a file.

Ls-1|wc-l 🡪Count the no of files & directories.

Metacharacter 🡪 used to o

* \* 🡪 current directory[matches 0 or more character]
* ? 🡪 previous directory[matches single character]
* [] 🡪(matches a set of character).

**Example :**

* Ls ?? 🡪 displays two character.
* Ls a\* 🡪 displays the file starting with “A”
* Ls f\*h 🡪 file starts with “F” & ends with “H”
* Ls f\*r\*h 🡪 file starts with “F”,& ends with “’h” & “R”in between the files
* Ls \*. 🡪 list all the files that has one character after dot[.]
* Ls [agr]\* 🡪it matches the files which are ls a\* ,ls g\* & ls \*r.
* Ls [h-k] 🡪 list the files that begins with h,i,j,k
* Rm \* .sh 🡪 remove all the files with .sh

**GREP :**

* The command which displays the line with matching pattern.
* It can be executed with more option.

**Syntax** :🡪 grep [option[s]] <pattern> <filename[s]>

|  |  |
| --- | --- |
| **OPTION** | **DEFINITION** |
| -i | Searches for both uppercase and lowercase characters |
| -l | Lists the names of files with matching lines |
| -n | Precedes each line with the relative line number in the file |
| -v | Inverts the search to display lines that do not match the  pattern |
| -c | Counts the lines that contain the pattern |
| -w | Searches for the expression as a complete word, ignoring  those matches that are substrings of larger words. |

* Grep -c 🡪 counts the no. of files that contains the pattern.
* Grep -h 🡪 omits the filename.
* Grep -i 🡪 ignore case.
* Grep -l 🡪 displays only the filename that contains the pattern.
* Grep -n 🡪 displays the line number that contains the pattern.
* Grep –v 🡪 displays all the lines that don’t not contain pattern.
* Grep - w 🡪 displays all the lines which is exactly matching the pattern.
* **Grep –c & grep-I**🡪can be used with many of the command[options]
* **Grep – s & Grep-w** 🡪 can be used with all the options
* Default internal field separater[**IFS**] 🡪 space.
* -d 🡪 refers the delimiter
* -f 🡪 refers to the field
* -c 🡪 to select the characters in each line.
* Tr 🡪translates & also helps in deleting the character.
* -who 🡪 displays all the details about the loggedin users information
* - who am I & Who are you 🡪 displays the user details.
* Finger 🡪 displays the **IDLE TIME** & description about the user.

Trainer : Mr.selvakumar date: 22-JUN-2015

Time : 10.10am-11.30am

Sort 🡪used to sort the file

Sort –r🡪 sort in the reverse order

Sort –u🡪 sort the whole file & displays only the unique lines

Cmp🡪compare the files.

Uniq🡪display the unique files & it will be counted only once.

* Uniq –c 🡪 displays the repetitive numbers.
* Uniq –d 🡪 displays only the repeated words.

Alias🡪 to have an another name for the file.

Script 🡪 records or stores all your files.

Du🡪displays the disk usage of the directories

Du \* 🡪 displays the disk usage all the files in directories

Du -s 🡪summarize disk usage of the directories

Df 🡪displays the disk free space of the file system

Set 🡪displays all the environment variable & its activities.

Set🡪Used to assign a positional parameters

* environment variable 🡪default variable which are assigned by login shell by inherit from its parent process

eval🡪evalutes the variable & perform its action

bc🡪used to do calculation[arithmetic]

file 🡪 displays the file type based on the content of a file.

Find🡪 used to search for a file based on size , filename, username, type, date modified.

* Find .-name x
* Find –regexp x
* Find –size –x
* Find .- type-x
* Find .-mtime -1

Stty🡪 disable input output components & also change terminal settings.

Ps🡪displays the status of active process.

Trainer : Mr.selvakumar date: 22-JUN-2015

Time : 4.35pm-6.00pm

**MODES :**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **MODES** | **KEYS** |
|  | command mode[standard] | esc key |
|  | insert mode | * i, 🡪insert before the cursor. * I 🡪insert from beginning of the current line. |
|  | append mode | * a🡪append after the cursor * A🡪appends at the end of the line |
|  | change mode | cc,c$, C |
|  | Replace mode | r🡪replace one character at current cursor position  R🡪 replace multiple characters[line] |
|  | Open mode | O 🡪creates a new line before & after the current line. |

Last line mode 🡪 substitute🡪save & write 🡪find.

* **Esc u** 🡪undo the file.
* **Esc .** 🡪redo
* J🡪 join the lines
* ~ or delete 🡪 change the case[upper & lower case vice versa..]

**Vi commands:**

* **:w**🡪 save the file
* **:w<filename>** 🡪 “save as” with different name.
* **:q**🡪quit after saving the file.
* **:q!** 🡪 quit without saving
* **:wq** 🡪 save & quit the file.[ZZ]
* **:n 🡪** opens the next file.[read multiple files & edit]
* **:r🡪** to insert the content of the another file.
* **:!** [command]**🡪**  used to execute unix commands
* **:r! 🡪** output of the unixcommand can be inserted into another file **.**

**Moving the cursor**

* **l🡪 to**  move towards right.
* **h🡪** to move towards left.
* **k 🡪** move to previous line.
* **j 🡪** move to next line.
* **0 🡪** go to the beginning of the current line.
* **$ 🡪** go to the end of the current line.
* **G 🡪** go to the beginning of the last line.
* **1G 🡪** go to the beginning of thefile.
* **10G 🡪**  to move to the 10 th line.
* **w🡪** move forward a word
* **b 🡪** to move backward.
* **e🡪**end of each word.

**MOVING THE SCREEN :**

* ^F🡪 move to the next screen
* ^B 🡪 move to the previous screen.
* ^D 🡪 move forward half of the screen[**customizable**]
* ^U🡪 move to previous half of the screen[**customizable**]

**DELETING OPTIONS:**

* x 🡪delete a current character[single]
* X🡪 deletes a previous character
* dd🡪to delete a line
* dw 🡪to delete a word
* db 🡪to delete a previous word
* d$ 🡪 deletes a character from the current cursor position to the end of the line.
* d0 🡪 deletes a character from the current cursor position to the beginning of the line.
* dG 🡪 deletes a character from the current cursor position to the end of the file.
* d1G 🡪 deletes a character from the current cursor position to the beginning of the file

**COPYING OPTIONS :**

* + **yy** 🡪copy the line.
  + **p**🡪to paste a line.
  + **yw** 🡪to copy a word.
  + **yb🡪** to copy a previous word.
  + **dd 🡪** to cut the line.

**CHANGING OPTIONS :**

* **cc🡪** to change the current line.
* **CC 🡪**To change the lines from current cursor position to the end of the line.
* **cw 🡪**To change the word.

**SUBSTITUTE OPTIONS:**

* + s🡪 substitute the current character.
  + S🡪 substitute the entire character.

Trainer : Mr.selvakumar date: 23-JUN-2015

Time : 10.20-11.30am

**SHELL SCRIPT :**

* Script 🡪A collection of command which are stored in a file.
* The shell can read this file & act on the commands as if they are typed on the keyboard.
* A script can be good for 🡪 a wide range of tasks that are **automated.**
* It will gather all the files & transmit them to a specific web server.
* Script act as a **BACKUP SERVER** on a network.
* Helps in unlocking the power of linux working machine.

**CREATING A SCRIPT :**

#!/bin/ksh

#first.sh

for file in \*

do

if grep –q POSIX $file

then

more $ file

fi

done

exit 0

* set command 🡪 used to display the environment variable and local variable.
* #!/bin/ksh 🡪should be at the top of the line so its is called **first line command.**

**Methods to execute :**

1. **Standard mode** :

Chmod +x Prog

./prog

1. **Normal mode:**

Ksh prog.

Csh prog.

Bash prog.

Sh prog.

**VARIABLE:**

**Global environment variable :**

The variables are set by the login shell new programs

shell inherits the environment of the parent shell.

Displayed in variable only in upper case[capital]

**Env 🡪** displays the environment variable.

PS 1🡪 [prompt string] default label is **$**

**Local variable :**

1)var = value

2) read variable name

3) positional parameter 🡪command line [co]

4) var =’cmd’

$ 🡪 used to access the value of the variable.

Echo 🡪 used to print a vriable similar to prinf

**Accessing the local variable :**

* Echo $ var
* Echo {}

Trainer : Mr.selvakumar date: 23-JUN-2015

Time : 3.15pm-pm

Meta characters 🡪 used to search a variable[file]

|  |  |
| --- | --- |
| **SPECIAL VARIABLES** | **FUNCTION** |
| $\* | All the arguments tat are passed during the run time |
| $# | Count the number of arguments |
| $@ |  |
| $? | Gives the **exit status** of the previously executed commands |
| $$ | Have the current parent [pid] |
| $0 | Similar to argv 0 |
| $1 | Similar to argv 1 |

* Shift 🡪used to shift the variables that helps in displaying the content which can display only 9 characters
* Shift ${10} 🡪 to display more than 9 characters[standard method]
* Single quote[‘’] 🡪 it will removes all the functionality of the metacharacters
* Double quotes[“”]🡪removes all the functionality of the metacharacters except in accessing the value of the variable.

|  |
| --- |
| * + Example 🡪 echo “$ fname” 🡪 displays raja as output |

* Back slash**[\]** 🡪 removes all the functionality of the metacharacters.
* ${varname}” ”🡪used to append a text.
* ${#company}🡪used to count the number of characters in a string in a variable.
* ${varname :- “ ”}🡪 if the variable is NULL.
* ${varname :+ “ ”}🡪 if the variable is not NULL.
* ${varname :? “ ”} 🡪 if the variable is NULL.
* Typeset 🡪 to assign a value to the variable.
  + - Typeset – i🡪declare a variable as integer data type
    - Typeset – r🡪 make the variable as read only
    - Typeset – l🡪convert into lowercase
    - Typeset – u🡪 convert into uppercase
* Unset 🡪 to remove the value of the variable
* Man🡪manual

**OPERATORS :**

* 🡪 -gt
* >= 🡪 -ge
* < 🡪 -lt
* <= 🡪 -le
* **==** 🡪 **-**eq
* != 🡪 -ne
* -a 🡪 AND
* -o 🡪 OR

**MULTIPLICATION :**

|  |
| --- |
| * Example **🡪**  result = ‘expr $num1 ‘\*’ $num2’ |

Trainer : Mr.selvakumar date: 25-JUN-2015

Time : 10.13-am

**SHELL SCRIPT Commands:**

* **-**F 🡪 used to check the input as regular file or not
* **-**d 🡪 used to check the input is directory or not
* -r 🡪 used to check the input is readable file or not
* -w🡪 used to check the input is writable file or not
* -x🡪 used to check the input is executable file or not
* -ss 🡪 used to check the input file size >0 or not
* -z🡪 used to check the input file size ==0 or not

**IF ELSE CONDITION :**

Syntax 🡪

* if test condition

then

…………….

……………

else

…………..

…………..

fi

* if [condition]

then

…………….

……………

else

…………..

…………..

fi

* if who/grep username

then

echo “”Welcome”

cat task list

else

echo “”Meet me”

fi

* if [-f ilename]

then

elif [-d filename]

then

echo “it is a directory”

ls filename

else

………..

fi

Trainer : Mr.Selvakumar

Date : 06-JUL2015 Time : 9.50-10.45a.m

**FOR LOOP:**

SYNTAX :

For varname in arg1 arg2 arg3……..argn

Do

……………….

………………. [commands /actions]

……………….

done

**EXAMPLE :**

For file in f1,txt f2,txt

Do

If[ -f $file]

Threw

More $file

Fi

done

**COLLECTIONS :**

1. Linked list
2. Array
3. Arraylist
4. Dictionary
5. Hashtable
6. Tree
7. Vector
8. Map
9. List

* FOR 🡪 used to access the limited elements[**known limits**]
* FOR EACH 🡪 used to access all the elements[**unknown limits**]

**EXAMPLE :**

For file in $\*

Do

If [-f $file]

Then

More $file

Else

Ls $file

Done

**WHILE LOOP:**

**SYNTAX:**

While [condition] <test conditon> <command>

Do

…………..

…………. [statements]

…………..

Done

* Executes when the condition is **true**
* Exits when the condition is **false**

**UNTIL LOOP:**

**SYNTAX:**

until [condition] <test conditon> <command>

Do

…………..

…………. [statements]

…………..

Done

* Executes when the condition is **false**
* Exits when the condition is **true**

**CASE :**

SYNTAX :

Case value in

Choice 1)

**……………**

**……………. [**actions/commands/ statments**]**

**;; (**break**)**

Choice 2)

**……………**

**……………. [**actions/commands/ statments**]**

**;; (**break**)**

\*

………………

esac

‘’[back quote] 🡪store the output of the command ina variable

Trainer : Mr.Selvakumar

Date : 09-JUL2015 Time : 9.50-10.45a.m

**EXAMPLE :**

* To find how many number of times the word echo is used.

Count=0

For value in ‘cat if.sh’

Do

If test $value =’echo’

Then

((count+=1))

Echo “$value”

Fi

Done

Echo”the no of echo is :$count”

Echo “----------using /$\*-------------”

For value in$\*

Do

Echo $value

Done

Echo “----------using /$\*-------------”

Do

Echo $value

Done

Echo “----------using /$@-------------”

Do

Echo $value

Done

Echo “----------using /$@-------------”

For value in $\*

Do

Echo $ value

done

* “$\*”🡪 printed as single line arguement
* “$@”🡪prints as entire argument

**EXAMPLE of WHILE LOOP:**

I=0

While [ $i -lt 5 ]

Do

Echo “$i”

i=expr $i +1

done

* who –q 🡪gives the username who are logged in the system

**ARRAY :**

* Set –A <name of array> elements of an array[DD DP PM QA TSG TR HR]
* City[0]=Madurai

**Accessing the elements of an array:**

* ${ City[0]}
* $ {city[\*]}🡪fetches all the elements of an array.
* ${# city[\*]}🡪 to count the number of elements in an array.
* Set –A dept $\* 🡪 to get the command line arguments.
* Shell is an command line interpreter.
* Interpreter checks line by line.

**DEFINE A FUNCTION:**

1. Display()

{

……..

…….commands

……

}

1. function Display()

{

……..

…….commands

……

}

* typeset 🡪used to declare a variable as a local variable
* -O 🡪 used to on the disable feature’
* Noclobber🡪overwrite cannot be done if the file exists already
* >> 🡪 used to overwrite
* Kill🡪 used to terminate a process
* Trap 🡪 signal handler command

Process:

It is a program in execution.

Job🡪 a unit of work

* Consists of commands specified in a single function

Types of jobs:

Foreground job

* A job that has a immediate attention
* User has to wait for a job to complete

Background job:

* The job that the user does not wait for
* It runs independently of user interaction

Unix shell allow users to :

* Make jobs execute in the background
* Move jobs from foreground to background
* Determine their status , & terminate them

Background jobs:

* Jobs that are run non-interactively
* Jobs that doesn’t require user command

SCHEDULING JOBS :

Many system administration tasks need to be performed, so that tasks can be done by scheduling to run daily using **cron** scheduling command.

Remember commands runs in batch:

* No output on terminal
* May get email
* Or just look for side effects

**SCHEDULING UTILITIES**

1. Crontab (= CRON Table)
   1. run a job based on a schedule
   2. job is executed on a periodic basis
2. at
   1. run a job some time in the future
   2. batch
   3. run a job when system load is low

**PERIODIC EXECUTION: CRONTAB**

* crontab is based on control file
* crontab file has 6 columns:

**minute hour day month weekday command**

* meaning:
  1. **minute 0-59**
  2. **hour 0-23**
  3. **day 1-31**
  4. **month 1-12**
  5. **weekday 1-7 (1=Mon,2=Tue, … ,7=Sun)**
  6. **command Any UNIX command**
* “\*” means any value[all the value

**CRONTAB COMMAND**

**options:**

-e to edit the control file

-l to list the control file

-r to remove the control file

* for superuser

-u to edit another user’s control file