USERS AND GROUPS

`useradd -D`	show default settings creation
`useradd -m test`	create user test with /home directory
`usermod`	user modification, you can do all with usermod
`usermod -G groupA userX`	add userX to groupA
`usermod -g groupA userX`	change the default group of userX to groupA
`groupmod -n groupA groupB`	change the name of groupA to groupB
`passwd`	change password
`chage`	change the password expiration date
`chfn`	change comment of user
`chsh`	change shell of user
`chpasswd < fileA`	read fileA that contains a list of user:passwd model to update there credentials

PERMISSIONS

`TYPES OF PERMISSIONS :`	Read=4, Write=2, eXecute=1
`SUID`	Set User ID on execution, s=4, if S = SUID set but no executable
`SGID`	Set Group ID on execution, s=2, all file create in this rep, get the same group of parent
`Sticky`	On directory,user who have group write permission can't delete other user file, like un /tmp
`COMMANDS :`	
`chown userA.groupA file`	change owner and group in same time of file
`chown .groupA file`	change group to groupA of file
`chgrp groupA file`	as the previous

JOURNALCTL COMMAND

`journalctl -list-boots`	list last boot log
`journalctl -b ID`	show boot log #ID
`journalctl -k`	show only kernel log
`journalctl PRIORITY`	show log by priority number 0-7
`journalctl -a -f`	show log in real time, -a all fields

PROCESSES INFORMATIONS

`ps aux`	show processes for a=all user, u=display processes user/owner, x=processes not attached to terminal
`ps -forest`	show processes in tree representation
`pas aux -sort=-rss`	sort +mem to -mem
`nice +5 updatedb &`	set priority en lunch the processus, more lower value = more %CPU
`renice -n -5 1010`	change priority of processus pid 1010 to 5

SYSADMIN 1

BASH ON CLI

help cmd	help on builtin command
type cmd	bash type of cmd
history	show command history
!5	show last 5 commands typed
!!	execute last command executed
!?str?	search last command who contains str and execute
source /.bashrc	read script and execute in current shell
cat file 2> out	stderr to out file
cat file &> out	stdout + stderr to out file
cat « TOP multiline TOP	push multiline text
cd -	go to last directory, cd $OLDPWD
(sleep 2; echo "Hi"; sleep 2	create a subshell
coproc MyCo{ sleep 2; }	subshell run in background

ENVIRONMENT VARIABLE

printenv	print global environment
env	run in specific environment or print environment
set	print global and local
export VAR	export VAR from local to global
unset	delete local variable
$HISTFILE	file that contains historic command, set to /dev/null to disable
$HISTSIZE	amount of command to memorize
$TMOUT	How long to autodisconnect shell
$RANDOM	get a random number

FILES

/etc/profile	executed on first login
/etc/bashrc	executed on bash launch
~/.bash_profile	executed on first bash launchf
~/.bash_logout	executed on every bash logout

SPECIAL VARIABLE

$[0-9]	get arguments from command line $0 $1 ...
$#	how many arguments

BUILTIN COMMAND

basename /usr/bin/bash	= bash
dirname /user/bin/bash	= /usr/bin
shift	shift argument, $2=$3, $3=$4 ...

REGULAR EXPRESSION

scr?pt	scrapt, script...
scr[ai]pt	scrapt, script
f[a-i]l	fell, fill, fall...
f[!a]ll	fell, fill... not fall
[abc]*	every word begin by A, B, C
{a,b,c}	a1 a2 a3 b1 b2 b3...
{a..f}{1..5}	a1 a2 a3 a4 a5 b1 b2...

SCRIPTING

$(cmd)	execute cmd
`cmd`	execute cmd
$[5 - 3]	compute
arr=(on two three)	an array
echo ${arr[1]}	get the 2nd item
echo ${arr[*]}	get all items
arr[2]=two	affect two to third item
read -p "Input :" valA valB	ask for values, and affect to valA valB
${var}	like $var
${var:-abcd}	if var is empty, affect abcd
var=/usr/bin/bash	use in next line :
${var#*/}	= usr/bin/bash
${var##*/}	= bash
${var%/*}	= /usr/bin
${var%%/*}	= empty
$((I++))	increment I

IF CONDITION

```
if [ $VAR -eq 1 ]; then
    echo "ok"
fi

if [ $STR = "hello" ]; then
    echo "yes"
elif [ $VAR != "bye" ]; then
    echo "yes yes"
else
    echo "no"
fi

[ -d "$dirpath" ] || mkdir $dirpath
[ -e "$dirpath" ] && echo "dir exist" || echo "dir not exist"
```

CASE CONDITION

```
case $(date +%a) in
    "dim.")
        make_backup.sh
    ;;
    "sam." | "mer.")
        empty_trash.sh
    ;;
    *)
    echo "do nothing"
esac
```

FOR LOOP

```
for NB in 0 1 2 3 4
do
    echo $NB
done
for FILE in $(ls -l $HOME)
do
    echo "$FILE"
done
```

WHILE LOOP

```
N=0
while [ $N -lt 10 ] ; do    # until pour l'inverse
    echo $N
    let N=$N+1
done
```