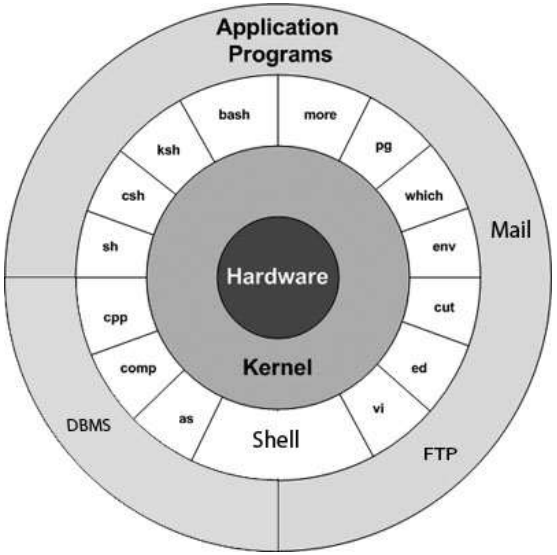
LINUX COMMANDS

Asist. Drd. Ing. Iuliana Marin

WHAT IS LINUX?

- The LINUX operating system is a set of programs that act as a link between the computer and the user. The computer programs that allocate the system resources and coordinate all the details of the computer's internals is called the operating system or kernel.
- Users communicate with the kernel through a program known as the shell. The shell is a command line interpreter; it translates commands entered by the user and converts them into a language that is understood by the kernel. Here is a basic block diagram of a LINUX system:



BASIC COMMANDS

- **Change password: $** passwd
- **Listing Directories and Files: $ ls OR ls -l** (long listing; displays information related to file permission => $ ls -l /home/iuliana); **-a** to list all files; $ ls -la

Note: enteries starting with **d.....** represent directories

- **To know the user: $** whoami
- **To know who is logged in to the computer at the same time**

  $ users (you see just their names)

  $ who OR w(give more details about the users, even their last entrance on your computer)
- **To logout:** $ logout
- **To shutdown a LINUX system**

  $ halt (Brings the system down immediately.)

  $ init 0 (Powers off the system using predefined scripts to synchronize and clean up the system prior to shutdown)

  $ init 6 (Reboots the system by shutting it down completely and then bringing it completely back up)

  $ poweroff (Shuts down the system by powering off.)

  $ reboot

  $ shutdown (usually from root)

LINUX - DIRECTORY MANAGEMENT

- Go to home directory: $ cd ~

Or to other user's home directory: $ cd ~username

Go to the last directory you accessed: $ cd -

Go to a specific directory: $ cd /usr/local/bin

From the above path, you can access the directory /usr/home/iuliana : $ cd ../../home/iuliana

- **Print current working directory:** $ pwd
- **List all files contained in /usr/local**: $ ls /usr/local
- **Create directory:** $mkdir mydir

Note: you can create more directories simultaneously

Sometimes the directories from the path do not exist, therefore use the -p option to the mkdir command: $mkdir -p /tmp/iuliana/test

- **Remove directories:** $rmdir dirname

Note: To remove a directory make sure it is empty which means there should not be any file or sub-directory inside this directory. You can also mose directories at the same time.

- **Rename directories:** $mv olddir newdir //move command

Note: The filename . (dot) represents the current working directory; and the filename .. (dot dot) represent the directory one level above the current working directory, often referred to as the parent directory.

LINUX FILE PERMISSION / ACCESS MODES

- **Owner permissions:** The owner's permissions determine what actions the owner of the file can perform on the file.
- **Group permissions:** The group's permissions determine what actions a user, who is a member of the group that a file belongs to, can perform on the file.
- **Other (world) permissions:** The permissions for others indicate what action all other users can perform on the file.
- Ex: $ ls –l /home/iuliana
- The first column represents different access modes: read (r), write (w), execute (x).
- The first three characters (2-4) represent the permissions for the file's owner. For example -**rwx**r-x**r--** represents that onwer has read (r), write (w) and execute (x) permission.
- The second group of three characters (5-7) consists of the permissions for the group to which the file belongs. For example -rwx**r-x**r-- represents that group has read (r) and execute (x) permission but no write permission.
- The last group of three characters (8-10) represents the permissions for everyone else. For example -rwxr-x**r--** represents that other world has read (r) only permission.

LINUX FILE PERMISSION / ACCESS MODES

- To change file or directory permissions, you use the **chmod** (change mode) command. There are two ways to use chmod: symbolic mode and absolute mode.
- Using chmod in Symbolic Mode:

Chmod operator	Description
+	Adds the designated permission(s) to a file or directory.
-	Removes the designated permission(s) from a file or directory.
=	Sets the designated permission(s).

- $ls -l testfile
- -rwxrwxr-- 1 iuliana users 1024 Oct 23 00:10 testfile
- $chmod o+wx testfile //o is for everyone else
- $ls -l testfile-rwxrwxrwx 1 iuliana users 1024 Oct 23 00:10 testfile
- $chmod u-x testfile
- $chmod g=r-x testfile
- $chmod o+wx,u-x,g=r-x testfile

USING CHMOD WITH ABSOLUTE PERMISSIONS

- The second way to modify permissions with the chmod command is to use a number to specify each set of permissions for the file.

Number	Octal Permission Representation	Ref
0	No permission	---
1	Execute permission	--x
2	Write permission	-w-
3	Execute and write permission: 1 (execute) + 2 (write) = 3	-wx
4	Read permission	r--
5	Read and execute permission: 4 (read) + 1 (execute) = 5	r-x
6	Read and write permission: 4 (read) + 2 (write) = 6	rw-
7	All permissions: 4 (read) + 2 (write) + 1 (execute) = 7	rwx

- $chmod 743 testfile
- $ls -l testfile-rwxr---wx 1 iuliana users 1024 Oct 23 00:14 testfile

CHANGING OWNERS AND GROUPS

- **chown:** The chown command stands for "change owner" and is used to change the owner of a file.
- **chgrp:** The chgrp command stands for "change group" and is used to change the group of a file.
- The chown command changes the ownership of a file. The basic syntax is as follows:
- $ chown user filelist
- **NOTE:** The super user, root, has the unrestricted capability to change the ownership of a any file but normal users can change only the owner of files they own.
- $ chgrp group filelist

LINUX - ENVIRONMENT

- Set a variables TEST and then we access its value using **echo** command.
- $TEST="LINUX Programming“ $echo $TEST => Result:LINUX Programming
- **For accessing them we use $sign as prefix. These variables retain their values until we come out of the shell**.
- The PATH variable specifies the locations in which the shell should look for commands. $PATH=/bin:/usr/bin Here each of the individual entries separated by the colon character, :, are directories.
- The characters that the shell displays as your command prompt are stored in the variable PS1. $PS1='=>‘ Your prompt would become =>.
- To set the value of PS1 so that it shows the working directory, issue the command: =>PS1="[\u@\h \w]\$“. The result of this command is that the prompt displays the user's username, the machine's name (hostname), and the working directory.
- Sequences that can be used as value arguments for PS1:

Escape Sequence	Description
\t	Current time, expressed as HH:MM:SS.
\d	Current date, expressed as Weekday Month Date
\n	Newline.
\s	Current shell environment.
\W	Working directory.
\w	Full path of the working directory.
\u	Current user.s username.
\h	Hostname of the current machine.
\#	Command number of the current command. Increases with each new command entered.
\$	If the effective UID is 0 (that is, if you are logged in as root), end the prompt with the # character; otherwise, use the $.

LINUX ENVIRONMENT

- When you issue a command that is incomplete, the shell will display a secondary prompt and wait for you to complete the command and hit Enter again.
- The default secondary prompt is > (the greater than sign), but can be changed by re-defining the **PS2** shell variable:
- $ echo "this is a
- > test“
- this is a
- test
- To redefine PS2 you can also use:
- $ PS2="secondary prompt->“
- $ echo "this is a
- secondary prompt->test“
- this is a
- test

ENVIRONMENT VARIABLES

Variable	Description
DISPLAY	Contains the identifier for the display that X11 programs should use by default.
HOME	Indicates the home directory of the current user: the default argument for the cd built-in command.
IFS	Indicates the Internal Field Separator that is used by the parser for word splitting after expansion.
LANG	LANG expands to the default system locale; LC_ALL can be used to override this. For example, if its value is pt_BR, then the language is set to (Brazilian) Portuguese and the locale to Brazil.
LD_LIBRARY_PATH	On many LINUX systems with a dynamic linker, contains a colon-separated list of directories that the dynamic linker should search for shared objects when building a process image after exec, before searching in any other directories.
PATH	Indicates search path for commands. It is a colon-separated list of directories in which the shell looks for commands.
PWD	Indicates the current working directory as set by the cd command.
RANDOM	Generates a random integer between 0 and 32,767 each time it is referenced.
SHLVL	Increments by one each time an instance of bash is started. This variable is useful for determining whether the built-in exit command ends the current session.
TERM	Refers to the display type
TZ	Refers to Time zone. It can take values like GMT, AST, etc.
UID	Expands to the numeric user ID of the current user, initialized at shell startup.

E.g.:
$ echo $HOME
/root

PRINTING FILES

- Many versions of LINUX include two powerful text formatters, **nroff** and **troff**.
- The **pr** command does minor formatting of files on the terminal screen or for a printer.
- pr option(s) filename(s)
- The **pr** changes the format of the file only on the screen or on the printed copy; it doesn't modify the original file.
- Some pr options:

Option	Description
-k	Produces k columns of output
-d	Double-spaces the output (not on all pr versions).
-h "header"	Takes the next item as a report header.
-t	Eliminates printing of header and top/bottom margins.
-l PAGE_LENGTH	Set the page length to PAGE_LENGTH (66) lines. Default number of lines of text 56.
-o MARGIN	Offset each line with MARGIN (zero) spaces.
-w PAGE_WIDTH	Set page width to PAGE_WIDTH (72) characters for multiple text-column output only.

PRINTING FILES

- Here are the contents of a sample file named nice
- $cat nice

Java
Apple

Let's use **pr** command to make a two-column report with the header *Likes*:

$pr -2 -h “Likes" nice
Oct 23 00:20 2014 Likes Page 1
Java Apple

- The command **lp** or **lpr** prints a file onto paper as opposed to the screen display.
- $ lp nice
- If you are using lp command, you can use -n**Num** option to print Num number of copies. Along with the command lpr, you can use **-Num** for the same.
- If there are multiple printers connected with the shared network, then you can choose a printer using -d**printer** option along with lp command and for the same purpose you can use -P**printer**option along with lpr command.
- The lpstat command shows what's in the printer queue: request IDs, owners, file sizes, when the jobs were sent for printing, and the status of the requests.
- Use lpstat -o if you want to see all output requests rather than just your own.
- $lpstat –o
- laserp-573 john 128865 Nov 7 11:27 on laserp
- The **lpq** gives slightly different information than lpstat -o:
- $lpq
- laserp is ready and printing
- Rank Owner Job Files Total Size
- active john 573 report.ps 128865 bytes

THE CANCEL AND LPRM COMMANDS

- The **cancel** terminates a printing request from the lp command. The **lprm** terminates lpr requests. You can specify either the ID of the request (displayed by lp or lpq) or the name of the printer.
- $cancel laserp-575
- request "laserp-575" cancelled
- To cancel whatever request is currently printing, regardless of its ID, simply enter cancel and the printer name:
- $cancel laserp
- request "laserp-573" cancelled
- The lprm command will cancel the active job if it belongs to you. Otherwise, you can give job numbers as arguments, or use a dash (-) to remove all of your jobs:
- $lprm 575
- The lprm command tells you the actual filenames removed from the printer queue.

SENDING AN EMAIL

- Syntax to send an email:
- $mail [-s subject] [-c cc-addr] [-b bcc-addr] to-addr

Option	Description
-s	Specify subject on command line.
-c	Send carbon copies to list of users. List should be a comma-separated list of names.
-b	Send blind carbon copies to list. List should be a comma-separated list of names.

- Send a test message to admin@yahoo.com:
- mail -s "Test Message" admin@yahoo.com
- You are then expected to type in your message, followed by an "control-D" at the beginning of a line. To stop simply type dot (.) as follows:
- Hi,
- This is a test
- .
- Cc:
- You can send a complete file using a redirect < operator as follows:
- $mail -s "Report 22/10/14" admin@yahoo.com < demo.txt
- To check incoming email :
- $mail
- no email

LINUX - PIPES AND FILTERS

- You can connect two commands together so that the output from one program becomes the input of the next program. Two or more commands connected in this way form a pipe.
- To make a pipe, put a vertical bar (|) on the command line between two commands.
- When a program takes its input from another program, performs some operation on that input, and writes the result to the standard output, it is referred to as a *filter*.
- The grep (g/re/p which means "globally search for a regular expression and print all lines containing it.“) program searches a file or files for lines that have a certain pattern.
- $grep pattern file(s)
- A regular expression is either some plain text (a word, for example) and/or special characters used for pattern matching.
- $ls -l | grep "Aug“
- -rw-rw-rw- 1 iuliana doc 11008 Aug 6 14:10 ch02
- There are various options which you can use along with grep command:

Option	Description
-v	Print all lines that do not match pattern.
-n	Print the matched line and its line number.
-l	Print only the names of files with matching lines (letter "l")
-c	Print only the count of matching lines.
-i	Match either upper- or lowercase.

- Let's use a regular expression that tells grep to find lines with "carol", followed by zero or more other characters abbreviated in a regular expression as ".*"), then followed by "Aug".
- Here we are using *-i* option to have case insensitive search:
- $ls -l | grep -i "carol.*aug“
- -rw-rw-r-- 1 carol doc 1605 Aug 23 07:35 macros

LINUX - PIPES AND FILTERS

- The **sort** command arranges lines of text alphabetically by default.
- $sort nice

Option	Description
-n	Sort numerically (example: 10 will sort after 2), ignore blanks and tabs.
-r	Reverse the order of sort.
-f	Sort upper- and lowercase together.
+x	Ignore first x fields when sorting.

- More than two commands may be linked up into a pipe.
- $ls -l | grep "Aug" | sort +4n
- This pipe sorts all files in your directory modified in August by order of size, and prints them to the terminal screen. The sort option +4n skips four fields (fields are separated by blanks) then sorts the lines in numeric order.
- To make it easier to read the sorted listing, pipe the output through **more** as follows:
- $ls -l | grep "Aug" | sort +4n | more

LINUX - PROCESSES MANAGEMENT

- When you execute a program on your LINUX system, the system creates a special environment for that program.
- The operating system tracks processes through a five digit ID number known as the **pid** or process ID . Each process in the system has a unique pid.
- When you start a process (run a command), there are two ways you can run it:
- Foreground Processes
- Background Processes
- By default, every process that you start runs in the foreground. It gets its input from the keyboard and sends its output to the screen.
- $ls ch*.doc
- This would display all the files whose name start with ch and ends with .doc.
- While a program is running in foreground and taking much time, we cannot run any other commands (start any other processes) because prompt would not be available until program finishes its processing and comes out.
- A background process runs without being connected to your keyboard. The simplest way to start a background process is to add an ampersand (&) at the end of the command.
- You need to know the job number to manipulate it between background and foreground.
- If you press the Enter key now, you see the following:
- [1] + Done ls ch*.doc &
- The first line tells you that the **ls** command background process finishes successfully. The second is a prompt for another command.

LINUX - PROCESSES MANAGEMENT

- See your own processes by running the **ps** (process status) comman: $ps
- One of the most commonly used flags for ps is the **-f** (f for full) option, which provides more information as shown in the following example:
- $ps –f
- UID PID PPID C STIME TTY TIME CMD
- iuliana 6738 3662 0 10:23:03 pts/6 0:00 first_one

Column	Description
UID	User ID that this process belongs to (the person running it).
PID	Process ID.
PPID	Parent process ID (the ID of the process that started it).
C	CPU utilization of process.
STIME	Process start time.
TTY	Terminal type associated with the process
TIME	CPU time taken by the process.
CMD	The command that started this process.

- There are other options which can be used along with **ps** command:

Option	Description
-a	Shows information about all users
-x	Shows information about processes without terminals.
-u	Shows additional information like -f option.
-e	Display extended information.

LINUX - PROCESSES MANAGEMENT

- Ending a process can be done in several different ways. Often, from a console-based command, sending a CTRL + C keystroke (the default interrupt character) will exit the command. This works when process is running in foreground mode.
- If a process is running in background mode then first you would need to get its Job ID using **ps**command and after that you can use **kill** command to kill the process as follows:
- $ps –f
- UID PID PPID C STIME TTY TIME CMD
- iuliana 6738 3662 0 10:23:03 pts/6 0:00 first_one
- $kill 6738
- Terminated
- If a process ignores a regular kill command, you can use **kill -9** followed by the process ID as follows:
- $kill -9 6738
- Terminated
- Each LINUX process has two ID numbers assigned to it: Process ID (pid) and Parent process ID (ppid). Each user process in the system has a parent process.
- Here is simple syntax to run top command and to see the statistics of CPU utilization by different processes:
- $top

LINUX - NETWORK COMMUNICATION UTILITIES

- When you work in a distributed environment then you need to communicate with remote users and you also need to access remote LINUX machines.
- The ping command sends an echo request to a host available on the network. Using this command you can check if your remote host is responding well or not.
- $ping hostname or ip-address
- Above command would start printing a response after every second. To come out of the command you can terminate it by pressing CTRL + C keys.
- $ping google.com
- PING google.com (74.125.67.100) 56(84) bytes of data.
- ftp stands for **F**ile **T**ransfer **P**rotocol. This utility helps you to upload and download your file from one computer to another computer.
- $ftp hostname or ip-address
- Above command would prompt you for login ID and password.
- $ftp something.com
- ftp> dir
- ftp> cd mpl
- ftp> get wave_shift
- ftp> quit

Command	Description
put filename	Upload filename from local machine to remote machine.
get filename	Download filename from remote machine to local machine.
mput file list	Upload more than one files from local machine to remove machine.
mget file list	Download more than one files from remote machine to local machine.
prompt off	Turns prompt off, by default you would be prompted to upload or download movies using mput or mget commands.
prompt on	Turns prompt on.
dir	List all the files available in the current directory of remote machine.
cd dirname	Change directory to dirname on remote machine.
lcd dirname	Change directory to dirname on local machine.
quit	Logout from the current login.

LINUX - THE VI EDITOR

- Screen-oriented text editor **vi**. This editor enable you to edit lines in context with other lines in the file.

Command	Description
vi filename	Creates a new file if it already does not exist, otherwise opens existing file.
vi -R filename	Opens an existing file in read only mode.
view filename	Opens an existing file in read only mode.

- $vi testfile
- A tilde (~) represents an unused line.
- To enter text, you must be in insert mode. To come in insert mode you simply type **i**. To get out of insert mode, press the **Esc** key, which will put you back into command mode.
- **Hint:** If you are not sure which mode you are in, press the Esc key twice, and then you'll be in command mode.
- The command to quit out of vi is :q. Once in command mode, type colon, and 'q', followed by return. If your file has been modified in any way, the editor will warn you of this, and not let you quit. To ignore this message, the command to quit out of vi without saving is **:q!**. This lets you exit vi without saving any of the changes.
- The command to save the contents of the editor is **:w**. You can combine the above command with the quit command, or :wq and return.
- The easiest way to save your changes and exit out of vi is the **ZZ** command. When you are in command mode, type ZZ and it will do the equivalent of :wq.
- You can specify a different file name to save to by specifying the name after the :w. For example, if you wanted to save the file you were working as another filename called filename2, you would type **:w filename2** and return.

LINUX - THE VI EDITOR

- To move around within a file without affecting your text, you must be in command mode (press Esc twice). Here are some of the commands you can use to move around one character at a time:

Command	Description
k	Moves the cursor up one line.
j	Moves the cursor down one line.
h	Moves the cursor to the left one character position.
l	Moves the cursor to the right one character position.

- There are many other ways to move within a file in vi. Remember that you must be in command mode (press Esc twice).
- To edit the file, you need to be in the insert mode.

Command	Description
x	Deletes the character under the cursor location.
X	Deletes the character before the cursor location.
dw	Deletes from the current cursor location to the next word.
d^	Deletes from current cursor position to the beginning of the line.
d$	Deletes from current cursor position to the end of the line.
D	Deletes from the cursor position to the end of the current line.
dd	Deletes the line the cursor is on.

- The vi has the capability to run commands from within the editor. To run a command, you only need to go into command mode and type **:!** command.

REGULAR EXPRESSIONS WITH SED

- A regular expression is a string that can be used to describe several sequences of characters.
- **sed** stands for **s**tream **ed**itor is a stream oriented editor which was created exclusively for executing scripts. Thus all the input you feed into it passes through and goes to STDOUT and it does not change the input file.
- Before we start, let us take make sure you have a local copy of /etc/passwd text file to work with **sed**.
- Sed can be invoked by sending data through a pipe to it as follows:
- $ cat /etc/passwd | sed
- The cat command dumps the contents of /etc/passwd to sed through the pipe into sed's pattern space. The pattern space is the internal work buffer that sed uses to do its work.
- The general syntax for sed: /pattern/action
- , **pattern** is a regular expression, and **action** is one of the commands given in the following table. If **pattern** is omitted, **action** is performed for every line as we have seen above.
- The slash characters (/) that surround the pattern are required because they are used as delimiters. **Deleting All Lines with sed:** $ cat /etc/passwd | sed 'd'

Range	Description
p	Prints the line
d	Deletes the line
s/pattern1/pattern2/	Substitutes the first occurrence of pattern1 with pattern2.

REGULAR EXPRESSIONS WITH SED

- The following command substitutes the first occurrence on a line of the string **root** with the string iuliana. $ cat /etc/passwd | sed 's/root/iuliana/‘. It is very important to note that sed substitutes only the first occurrence on a line. If the string root occurs more than once on a line only the first match will be replaced.
- To tell sed to do a global substitution, add the letter **g** to the end => ‘s/root/iuliana/g’.

NAVIGATING THE FILE SYSTEM

Command	Description
cat filename	Displays a filename.
cd dirname	Moves you to the directory identified.
cp file1 file2	Copies one file/directory to specified location.
file filename	Identifies the file type (binary, text, etc).
find filename dir	Finds a file/directory.
head filename	Shows the beginning of a file.
less filename	Browses through a file from end or beginning.
ls dirname	Shows the contents of the directory specified.
mkdir dirname	Creates the specified directory.
more filename	Browses through a file from beginning to end.
mv file1 file2	Moves the location of or renames a file/directory.
pwd	Shows the current directory the user is in.
rm filename	Removes a file.
rmdir dirname	Removes a directory.
tail filename	Shows the end of a file.
touch filename	Creates a blank file or modifies an existing file.s attributes.
whereis filename	Shows the location of a file.
which filename	Shows the location of a file if it is in your PATH.

NAVIGATING THE FILE SYSTEM

- The first way to manage your partition space is with the df (disk free) command. The command df -k (disk free) displays the disk space usage in kilobytes, as shown below:
- $df –k
- Filesystem 1K-blocks Used Available Use% Mounted on
- /dev/vzfs 10485760 7836644 2649116 75% /
- /devices 0 0 0 0% /devices
- The du (disk usage) command enables you to specify directories to show disk space usage on a particular directory. –h is for human readable.
- $du –h /etc
- 10k /etc/cron.d
- A file system must be mounted in order to be usable by the system. To see what is currently mounted (available for use) on your system, use this command: $ mount
- The /mnt directory, by Linux convention, is where temporary mounts (such as CD-ROM drives, remote network drives, and floppy drives) are located. If you need to mount a file system, you can use the mount command with the following syntax:
- mount -t file_system_type device_to_mount directory_to_mount_to
- $ mount -t iso9660 /dev/cdrom /mnt/cdrom
- To unmount (remove) the file system from your system, use the **umount** command by identifying the mountpoint or device
- For example, to unmount cdrom, use the following command:
- $ umount /dev/cdrom

USER ADMINISTRATION

- groupadd [-g gid [-o]] [-r] [-f] groupname
- $ groupadd developers
- $ groupmod -n new_modified_group_name old_group_name
- $ groupmod -n developer developer_2
- create a user's account:
- useradd -d homedir -g groupname -m -s shell -u userid accountname
- Following is the example which would create an account *mcmohd* setting its home directory to/*home*/*mcmohd* and group as *developers*.
- $ useradd -d /home/mcmohd -g developers -s /bin/ksh mcmohd
- Once an account is created you can set its password using the **passwd** command: $ passwd mcmohd
- The **usermod** command enables you to make changes to an existing account from the command line. It uses the same arguments as the useradd command, plus the -l argument, which allows you to change the account name.
- For example, to change the account name *mcmohd* to *mcmohd20* and to change home directory accordingly, you would need to issue following command:
- usermod -d /home/mcmohd20 -m -l mcmohd mcmohd20
- The **userdel** command can be used to delete an existing user. This is a very dangerous command if not used with caution.
- There is only one argument or option available for the command: .r, for removing the account's home directory and mail file.

$ userdel -r mcmohd20

Command	Description
useradd	Adds accounts to the system.
usermod	Modifies account attributes.
userdel	Deletes accounts from the system.
groupadd	Adds groups to the system.
groupmod	Modifies group attributes.
groupdel	Removes groups from the system.

SYSTEM PERFORMANCE

Command	Description
nice/renice	Run a program with modified scheduling priority
netstat	Print network connections, routing tables, interface statistics, masquerade connections, and multicast memberships
time	Time a simple command or give resource usage
uptime	System Load Average
ps	Report a snapshot of the current processes.
vmstat	Report virtual memory statistics
gprof	Display call graph profile data
prof	Process Profiling
top	Display system tasks

DOS COMMANDS FROM A JAVA PROGRAM

- You can't run DOS commands directly by specifying Runtime.getRuntime().exec(dosCommand) in Java.
- Instead, to execute a DOS command (such as DIR, or RD) from a Java program, you need to prepend the command to run the Windows command shell cmd /c to the command you want to execute. The /c switch terminates the command shell after the desired command completes.
- The syntax: Runtime.getRuntime().exec("cmd /c " + dosCommand);

WINDOWS DOS COMMANDS IN JAVA

Useful commands: http://ss64.com/nt/

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class ShellTest {
    public static void main(String[] args) throws java.io.IOException, java.lang.InterruptedException {
        //execute a Windows command to list content of the Documents folder
        String commandArray[] = {"cmd", "/c", "dir", "C:\\Users\\Iuliana\\Documents"};
        try {
            //Process process = Runtime.getRuntime().exec(command);
            Process process = Runtime.getRuntime().exec(commandArray2);
            BufferedReader reader = new BufferedReader(
                    new InputStreamReader(process.getInputStream()));
            String line;
            while ((line = reader.readLine()) != null) {
                System.out.println(line);
            }
            reader.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

EXERCISES

1) Run some commands from the PPT on your virtual machine. (2 points)
2) Ping the address www.google.com. (1 point)
3) Create the folder Music in your folder Documents of your computer. (1 point)

Hint: put the two paths as separate strings

4) Copy a file to another location. (1 point)
5) Copy a directory from one source to a new destination. (1 point)
6) Delete a file. (1 point)
7) Delete a directory and all its subdirectories and files that are inside it. (1 point)
8) Move a file to a folder to another. (1 point)
9) Rename your moved file. (1 point)