Basic UNIX Command Line (shell) navigation

1- What is Unix?

Pronounced yoo-niks, a popular multi-user, multitasking operating system developed at Bell Labs in the early 1970s. Created by just a handful of programmers, UNIX was designed to be a small, flexible system used exclusively by programmers.

UNIX was one of the first operating systems to be written in a high-level programming language, namely C. This meant that it could be installed on virtually any computer for which a C compiler existed. This natural portability combined with its low price made it a popular choice among universities. (It was inexpensive because antitrust regulations prohibited Bell Labs from marketing it as a full-scale product.)

Bell Labs distributed the operating system in its source language form, so anyone who obtained a copy could modify and customize it for his own purposes. By the end of the 1970s, dozens of different versions of UNIX were running at various sites.

2- Directories:

File and directory paths in UNIX use the forward slash "/" to separate directory names in a path.

Examples:

/	"root" directory
/usr	directory usr (sub-directory of / "root" directory)
/usr/STRIM100	STRIM100 is a subdirectory of /usr

3- Moving around the file system:

pwd	Show the "present working directory", or current directory.
cd	Change current directory to your HOME directory.
cd /usr/STRIM100	Change current directory to /usr/STRIM100.
cd INIT	Change current directory to INIT which is a sub-directory of the current directory.
cd ..	Change current directory to the parent directory of the current directory.
cd ~bob	Change the current directory to the user bob's home directory (if you have permission).

4- Listing directory contents:

ls	list a directory
ls –l	list a directory in long (detailed) format

Example:

```
$ ls –l
-rw-r- -r- - 1 cliff        user      767392   Jun 6  14:28    scanlib.tar.gz
^ ^    ^    ^  ^   ^            ^             ^          ^          ^              ^
| |    |    |  |   |            |             |          |          |              |
| |    |    |  |   owner   group       size       date      time        name
| |    |    |  number of links to file or directory contents
| |    |  permissions for world
| |    permissions for members of group
| permissions for owner of file: r = read, w = write, x = execute -=no permission
```

type of file: - = normal file, d=directory, l = symbolic link, and others...

ls -a List the current directory including hidden files. Hidden files start with "."

ls -ld * List all the file and directory names in the current directory using long format. Without the "d" option, ls would list the contents of any sub-directory of the current. With the "d" option, ls just lists them like regular files.

5- Understanding pathnames

Suppose you have a directory named backups under path ~/ unixstuff
First type cd to get back to your home-directory, then type

```
% ls unixstuff
```

to list the conents of your unixstuff directory.
Now type

```
% ls backups
```

You will get a message like this -

```
backups: No such file or directory
```

The reason is, backups is not in your current working directory. To use a command on a file (or directory) not in the current working directory (the directory you are currently in), you must either cd to the correct directory, or specify its full pathname. To list the contents of your backups directory, you must type

```
% ls unixstuff/backups
```

6- Changing file permissions and attributes

chmod 755 file Changes the permissions of file to be rwx for the owner, and rx for the group and the world. (7 = rwx = 111 binary. 5 = r-x = 101 binary)

chgrp user file Makes file belong to the group user.

chown cliff file Makes cliff the owner of file.

chown -R cliff dir	Makes cliff the owner of dir and everything in its directory tree.

You must be the owner of the file/directory or be root before you can do any of these things.

7- Moving, renaming, and copying files:

cp file1 file2	copy a file
mv file1 newname	move or rename a file
mv file1 ~/AAA/	move file1 into sub-directory AAA in your home directory.
rm file1 [file2 ...]	remove or delete a file
rm -r dir1 [dir2...]	recursively remove a directory and its contents
mkdir dir1 [dir2...]	create directories
mkdir -p dirpath	create the directory dirpath, including all implied directories in the path.
rmdir dir1 [dir2...]	remove an empty directory

8- Viewing and editing files:

cat filename	Dump a file to the screen in ascii.
more filename	Progressively dump a file to the screen: ENTER = one line down SPACEBAR = page down q=quit
less filename	Like more, but you can use Page-Up too. Not on all systems.
vi filename	Edit a file using the vi editor. All UNIX systems will have vi in some form.
emacs filename	Edit a file using the emacs editor. Not all systems will have emacs.
head filename	Show the first few lines of a file.
head -n filename	Show the first n lines of a file.
tail filename	Show the last few lines of a file.
tail -n filename	Show the last n lines of a file.

9- Environment variables

You can teach your shell to remember things for later using environment variables. Example:

export CASROOT=/usr/local/CAS3.0	Defines the variable CASROOT with the value /usr/local/CAS3.0.

By prefixing $ to the variable name, you can evaluate it in any command:

cd $CASROOT	Changes your present working directory to the value of CASROOT

10-Searching for strings in files: The grep command

grep string filename	prints all the lines in a file that contain the string

The grep command is case sensitive; To ignore upper/lower case distinctions, use the -i option, i.e. type

```
% grep –i
```

11-Searching for files : The find command

find search_path -name filename

find . -name aaa.txt	Finds all the files named aaa.txt in the current directory or any subdirectory tree.
find / -name vimrc	Find all the files named 'vimrc' anywhere on the system.
find /usr/local/games -name "*xpilot*"	Find all files whose names contain the string 'xpilot' which exist within the '/usr/local/games' directory tree.

12-Word count:

A handy little utility is the wc command, short for word count. To do a word count on science.txt, type

```
% wc -w science.txt
```

To find out how many lines the file has, type

```
% wc -l science.txt
```

13-Redirection:

grep string filename > newfile	Redirects the output of the above grep command to a file 'newfile'.
grep string filename >> existfile	Appends the output of the grep command to the end of 'existfile'.

The redirection directives, > and >> can be used on the output of most commands to direct their output to a file.

14-Pipes:

The pipe symbol "|" is used to direct the output of one command to the input of another.

Example:

ls -l | more	This commands takes the output of the long format directory list command "ls -l" and pipes it through the more command (also known as a filter). In this case a very long list of files can be viewed a page at a time.

15-Command Substitution

You can use the output of one command as an input to another command in another way called command substitution. Command substitution is invoked when by enclosing the substituted command in backwards single quotes.

Example:

```
cat `find . -name aaa.txt`
```

which will cat (dump to the screen) all the files named aaa.txt that exist in the current directory or in any subdirectory tree.

16-Basics of the vi editor

Opening a file

vi filename

Creating text

Edit modes: These keys enter editing modes and type in the text of your document.

i	Insert before current cursor position
I	Insert at beginning of current line
a	Insert (append) after current cursor position
A	Append to end of line
r	Replace 1 character
R	Replace mode
<ESC>	Terminate insertion or overwrite mode

Deletion of text

x	Delete single character
dd	Delete current line and put in buffer
ndd	Delete n lines (n is a number) and put them in buffer
J	Attaches the next line to the end of the current line (deletes carriage return).

Oops

u	Undo last command

cut and paste

yy	Yank current line into buffer
nyy	Yank n lines into buffer
p	Put the contents of the buffer after the current line
P	Put the contents of the buffer before the current line

cursor positioning

^d	Page down
^u	Page up
:n	Position cursor at line n
:$	Position cursor at end of file
^g	Display current line number

h,j,k,l Left,Down,Up, and Right respectivly. Your arrow keys should also work if if your keyboard mappings are anywhere near sane.

Saving and quitting and other "ex" commands

These commands are all prefixed by pressing colon (:) and then entered in the lower left corner of the window. They are called "ex" commands because they are commands of the ***ex*** text editor **-** the precursor line editor to the screen editor vi. You cannot enter an "ex" command when you are in an edit mode (typing text onto the screen)
Press <ESC> to exit from an editing mode.

:w	Write the current file.
:w new.file	Write the file to the name 'new.file'.
:w! existing.file	Overwrite an existing file with the file currently being edited.
:wq	Write the file and quit.
:q	Quit.
:q!	Quit with no changes.
:e	filename Open the file 'filename' for editing.

References:

1- http://freeengineer.org/learnUNIXin10minutes.html
2- http://linux-training.be/files/books/LinuxFun.pdf