$ WACM Presents: Intermediate Linux

You can find our Basic Linux tutorial here and this tutorial at https://goo.gl/1YX344
[1] $ ssh jalal@best-linux.cs.wisc.edu

wget (web get, downloading something from web)
[78] $ wget https://github.com/lamiastella/WACM/archive/master.zip
[78] $ unzip master.zip
[78] $ mv WACM-master WACM

Some Vim magics:
/* check **$ vimtutor**
vimrc settings:
[2] $ vi ~/.vimrc
press **i** and write the following in the editor:
syntax on
set number
set ignorecase
set cindent
type **:wq** to save the file
Commenting something in **~/.vimrc** is by putting a double quote (**"**) in the beginning of line
" set cindent
Do the following to auto-indent the entire code within Vim:
[3] $ vim non_indented.c
copy the following:

```
fun() {

for()

{

for()

{

if()

{

}

}

}

}
```

Go to command mode (by pressing **Esc**) and type **gg=G**, voila!

If you want to split your Vim editor horizontally or vertically use the following commands:

:vsp

:sp
to move to the right/left/up/down enter:
Ctrl+w and then **l/r/k/j** respectively
To give each of the splits a name use:
:e desired_name.txt

ls (Listing files and directories)
[4] $ ll (equal to ls -alF)
[5] $ ls -ltr
[6] $ ls -la

- -a option is to show hidden files (will show . and .. too)
- -l option is to show the output as a long list along with various attributes e.g. permissions, file sizes, use, group, modification time etc.
- -F will append one of */=>@| to the entries, it is basically used to differentiate files from directories as it will append / to the directory entries
- -t option will sort the entries by modification date (with newest first)
- -r will reverse the sorting order.

> (redirecting output to another file/stream):

[7] $ ls -l > ls-l.txt

clear (Clearing the screen)
[8] $ clear

bash scripts (automating things using a series of Linux commands)
[9] $ which bash
[10] $ vi simple_bash.sh
add the following to the file and save it:
#!/bin/bash
echo "hello, $USER. I wish to list some files of yours"
echo "listing files in the current directory, $PWD"
ls # list files
Execute it using:
[11] $ sh simple_bash.sh
OR
[12] $ bash simple_bash.sh
OR make your bash script executable using chmod command:
[13] $ chmod +x simple_bash.sh
and run it using ./ before the name of the .sh file:

[14] $ ./simple_bash.sh

df (Checking your disk space utilization)

[15] $ df

the above command but with h flag for human-readable format:

[16] $ df -h

Display the information on home directory:

[17] $ df -hT /home

```
[jalal@adelie-07] (68)$ df -hT /u/j/a/jalal/WACM
Filesystem     Type  Size  Used Avail Use% Mounted on
AFS            afs   2.0T     0  2.0T   0% /afs
```

Display the file system types using T flag:

```
[jalal@adelie-07] (73)$  df -T
Filesystem     Type   1K-blocks     Used  Available Use% Mounted on
/dev/sda3      ext4    41395504 11669032   27617032  30% /
tmpfs          tmpfs    8125724     7108    8118616   1% /dev/shm
/dev/sda1      ext4     3997376   130572    3657092   4% /boot
/dev/sda7      ext4   123970516  1076408  116590108   1% /tmp
/dev/sda4      ext4     5029504    75828    4691532   2% /var/home
/dev/sda6      ext4      999320   385500     561392  41% /var/tmp
/dev/sda5      ext4     3997376     8184    3779480   1% /var/vice/cache
AFS            afs   2147483647        0 2147483647   0% /afs
```

ps (Viewing processes running on the system)

[18] $ ps

[19] $ ps aux

[20] $ ps -ef

[21] $ ps aux --sort=-pcpu,+pmem

kill

[22] $ kill 30747

[23] $ kill -9 30735

grep (general regular expression parser)

[24] $ cd ~/WACM

[25] $ ls -ltr

[26] $ vi users.txt

[27] $ tail -5 users.txt

[28] $ head -5 users.txt

[29] $ grep -i joe users.txt

[30] $ grep joe users.txt

[31] $ grep ^Joe users.txt

[32] $ grep "Mona" *users*

[33] $ grep -n "Mona" *users*

[34] $ grep -r "Mona" *
[35] $ grep --color -r "Mona" *
Use “egrep” to search for multiple words within the same file:
[36] $ egrep --color -w -n "Mona|Sejal" good_users.txt
Count the number of times an item is found with -c flag:
[37] $ egrep -c "Mona|Sejal" good_users.txt
Use the -v flag to print everything else except the words you are looking for:
[38] $ egrep --color -w -v "Mona|Sejal" good_users.txt
[39] $ ps ax | grep jalal
List your PCI devices:
[40] $ lspci
List your USB devices:
[41] $ lsusb
[42] $ lspci | grep Eth
[43] $ lspci | grep VGA
You can show your CPU model using grep like the following:
[44] $ cat /proc/cpuinfo | grep -i 'Model'
^Students also try only “**$ cat /proc/cpuinfo**”

uname (Unix name)
[45] $ uname -o (Operating system)
[46] $uname -p (Processor type)
[47] $ uname -n (Node name)
[48] $ uname -s (Kernel name)
[49] $ uname -a (Displaying all the information)
[50] $ uname -r (Display kernel version)

sort
[51] $ cat users.txt
[52] $ sort users.txt
[53] $ sort users.txt > sorted_user.txt
[54] $ cat sorted_user.txt
[55] $ sort -r users.txt > reverse_sorted_user.txt
[56] $ cat reverse_sorted_user.txt

ln -s (to create a soft link)
Types of links: *soft(symbolic) link *hard link
ln -s {target-filename} {symbolic-filename}
[57] $ mkdir soft_link
[58] $ cd soft_link/
[59] $ touch test
[60] $ cd ..
[61] $ ln -s soft_link/test .

[62] $ ls -l

chmod (For changing access permissions)
[63] $ chmod 777 users.txt
[64] $ ls -l users.txt
[65] $ chmod 644 users.txt
[66] $ ls -l users.txt

Reference	Class	Description
u	user	the owner of the file
g	group	users who are members of the file's group
o	others	users who are neither the owner of the file nor members of the file's group
a	all	all three of the above, same as ugo

[67] $ chmod ug=rx users.txt
[68] $ ls -l users.txt
[69] $ chmod u-g users.txt
[70] $ ls -l users.txt

vimdiff/diff (showing differences in two (vim) files):
[71] $ chmod 777 users.txt
[72] $ vimdiff users.txt sorted_user.txt
[73] $ diff users.txt sorted_user.txt

curl (For URL manipulation and transfer)
[74] $ curl https://www.cs.wisc.edu/ > cs.html
[75] $ curl -O https://www.cs.wisc.edu/
[76] $ curl -O http://research.cs.wisc.edu/bullying/agreement.txt
[77] $ curl -o cs_web https://www.cs.wisc.edu/

which/whereis/whatis
[79] $ which gcc
[80] $ whereis gcc
[81] $ whatis jhat
[82] $ whatis gcc

awk (Aho, Weinberger, and Kernighan)
Printing only third and fourth column of a document:
[83] $ awk '{print $3 "\t" $4}' marks.txt
Printing every line that has the pattern A in it:

[84] $ awk '/A/' marks.txt
Printing columns that have a specific pattern in their line:
[85] $ awk '/a/ {print $3 "\t" $4}' marks.txt
Print every line with length more than 20 characters:
[86] $ awk 'length($0) > 20 ' awk_example.txt
Print the entire line:
[87] $ echo "hello1 hello2 hello3" | awk ' {print $0} '
Print the third entry in the line:
[88] $ echo "hello1 hello2 hello3" | awk ' {print $3} '
Or maybe you were wondering how to print hi 28 times?
[89] $ yes | head -28 | awk '{ print "hi" }'
Or maybe 10 random integers modulo 7?
[90] $ yes | head -10 | awk '{print int(100*rand()) % 7}'

sed (Short for "stream editor", allows you to filter and transform text)
Use the following command for replacing all the unix words with linux:
[91] $ sed 's/unix/linux/' sed_example.txt
[92] $ sed 's/7.30/6.30/' sed_example.txt > songs.txt
Or replace only the n-th occurrences of a word with another word:
[93] $ sed 's/unix/linux/2' file.txt
Seeing the lines containing the pattern John:
[94] $ sed -n '/John/p' sed_example.txt

xargs (Processes the standard input on all unix flavoured operating systems)
Displaying only 3 items per line:
[95] $ echo a b c d e f| xargs -n 3
Find all the .C files in the current directory and delete them:
[96] $ touch mona.c |find . -name "*.c" -type f -print | xargs /bin/rm -f
create the file named “not important_file”:
[97] $ touch 'not important_file'
Find and remove all the files with pattern not:
[98] $ find . -name not* -print0 | xargs -0 rm
Find everything in my home directory with wide-open permissions:
[99] $ find ~ -perm 777
tar (compressing/decompressing files):
For creating .tar.gz archive of the current directory you should use the z flag after the tar command to create a gzip compressed file:
[100] $ tar cvzf current_directory.tar.gz .
You should use the j flag after the tar command to create a bz2 compressed file:
[101] $ tar cvfj current_directory.tar.bz2 .
You should use the x flag for untarring and also C flag if you want to untar in a specific directory:
[102] $ mkdir untar

[103] $ tar -xvf current_directory.tar.gz -C untar/
use the t flag to browse the list of files in the tar file:
[104] $ tar -tvf current_directory.tar.bz2
For adding a single file to a tar.gz file, first extract it to a tar file using gunzip command and update the tar file using uf flag. Eventually compress it back to the same tar.gz file using gzip:
[105] $ tar cvzf tar_new.tar.gz .
[106] $ gunzip tar_new.tar.gz
[107] $ ls -ltr
[108] $ tar -uf tar_new.tar new_file
[109] $ gzip tar_new.tar
[110] $ ls -ltr
[111] $ tar -tvf tar_new.tar.gz

Zip (Compressing/decompressing files to/of zip format)
Compress all your files in the current directory to a zip file:
[112] $ zip -r myfiles.zip *
Unzip all your files in a zip file to a destination folder:
[113] $ unzip myfiles.zip -d unzip

Some Network Commands:
[114] $ traceroute -I google.com
[115] $ traceroute -U google.com
[116] $ tracepath google.com
[117] $ whois google.com
[118] $ whoami
[119] $ hostname
[120] $ dig google.com
[121] $ ping google.com
[122] $ host google.com

ltrace/strace
ltrace (It traces the calls to the library function. It executes the program in that process.You can also note that ltrace gives the results in the order the functions are called in the program.)
[123] $ ltrace /usr/bin/who
strace (strace command is used to trace the system calls made by the program. If a program is not using any library function, and it uses only system calls, then using plain ltrace, we cannot trace the program execution.)
[124] $ strace /usr/bin/who

MORE FUN THO:
[125] $ telnet towel.blinkenlights.nl

[126] $ telnet nyancat.dakko.us
[127] $ aafire
[128] $ cal
[129] $ factor 210000
and bad.horse of course:
[130] $ traceroute bad.horse

Git commands: (Create a git account at https://github.com and then create a repository)
[134] $ mkdir test_git
[135] $ cd test_git
[136] $ git init
replace lamiastella with your github username and WACM with your repo's name:
[137] $ git remote add origin git@github.com:lamiastella/WACM.git
[138] $ touch test
[139] $ git add .
[140] $ git status
[141] $ git commit -m "adding the test to the repo WACM"
[142] $ git pull origin master
[143] $ git push -u origin master

Check wireless signal strength
[131] $ iwconfig - link quality and signal level values
[132] $ watch -n1 iwcofig (updates the information every 1s, walk around with the laptop)
[133] $ watch -n1 "awk 'NR==3 {print \"Wifi Signal Strength = \" \$3 \"00 %\"}'" /proc/net/wireless"

chown (Change ownership of a directory)
[144] $ chown -R 755 .
Least but not last for some commands you need to be a sudoer! and if you are not in the sudoer list in CSL you'll see this:

```
[sudo] password for jalal:
jalal is not in the sudoers file.  This incident will be reported.
```

Try these commands at home:
[145] $ echo $PATH
[146] $ screen
[147] $ top
[148] $ lsof
[149] $ nslookup google.com
[150] $ shutdown -r now
and many more from the following links:
List of Linux commands:https://en.wikipedia.org/wiki/List_of_Unix_commands
Most voted Linux commands:http://www.commandlinefu.com/commands/browse/sort-by-votes