1 Introduction Linux

Linux is an open-source operating system (OS) developed based on the kernel created by Linus Torvalds. In the last two decades, Linux has gained so much popularity and now it is used on many platforms. Nowadays, most of the high-end servers to mobile phones (Android OS or iOS) run on different variants of Linux.

Generally, Linux computers/servers are installed for multi-user usage. However, in this bioinformatics course, we will be working on a local Linux machine. All the commands what you learn here can be used in any distribution (i.e. Ubuntu, CentOS, Debian, SuSE, etc) of Linux operating system.

1.1 The Terminal

We use terminal (AKA command line interface) to interact with the operating system. The terminal by default runs one of the "shells". Shell is a program that sits between the user and the kernel and translates user commands into machine code. The advantages of using command line are greater control and flexibility over the system or software. Moreover, multiple commands can be saved in a file and executed as a script.

The most common shells are:

- Bourne Shell
- Bourne Again Shell (variant is Z Shell)
- C Shell (variant is T Shell)
- K Shell

Among these Bourne Again Shell (BASH) is the most popular one. This is the default shell on the system and we will be using it throughout our course.

1.2 Connecting to Linux Server

If you are going to work on a remote server use "ssh" command to connect to it.

```
ssh user@ServerName
ssh user@ipaddress
```

There are other commands to connect to the server (telnet, rsh etc...) which are unsecure and outdated. However, in this course we will be using locally installed Linux system. Please use provided username and password to enter into your account and open "Terminal".

1.3 Linux command structure

When you open a terminal, you will see command prompt ready to take commands. The default location on the terminal is your "home directory". It is represented with ~ (tilde) symbol.

All Linux commands are single words (can be alpha-numeric), with optional parameters followed by arguments. For historical reasons, some of the early commands are only two letter long and case sensitive. Most of the command options (also called flags) are single letters. They should be specified after the command before giving any input.

```
ls -l ~/Data-Sreenu

total 20
drwxr-x--- 2 manager manager 4096 Feb 20 12:21  exFiles
drwxr-x--- 2 manager manager 4096 Mar  3 18:06  FQs
drwxr-x--- 2 manager manager 4096 Feb 20 12:21  GFF
drwxr-x--- 2 manager manager 4096 Feb 20 12:21 'HCV genotypes'
drwxr-x--- 2 manager manager 4096 Feb 20 12:21  Ref
```

Here "ls" is the command to list the contents of the directory, "-l" is the option for long listing and "~/Data-Sreenu" is the input, which is optional in this case. Without the input, "ls" shows the contents of the current directory.

Please remember:

- Linux commands are case sensitive
- Single words
- Options have to follow the command
- Options can start with a single hyphen and a character or a double hyphen and a word

- Single character options can be combined
- Argument can be one or multiple inputs (ls -l Documents Desktop)
- You can write more than one command separating with a semicolon;

You can use "tab" to auto-fill the command.

1.4 First Commands

`ls`
(You might see a different output than shown below...)

```
 Lists information about the files/directories. Default is current directory.
Sorts entries alphabetically.

Commonly used options:
-l long list
-a show all files (including hidden files)
-t sort based on last modified time

Ex:
ls -l

drwxrwxr-x 31 training training   4096 Dec 19 13:15 anaconda2
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
-rw-rw-r--  1 training training 669825 Jan 18 13:02 bamtools-master.zip
drwxrwxr-x  3 training training   4096 Jan 29 14:22 bin
drwxr-xr-x  2 training training   4096 Dec 18 20:19 Desktop
drwxr-xr-x  2 training training   4096 Dec 18 20:19 Documents
```

Here Left to right columns are:

- File permissions
- Number of links
- Owner's name
- Group's name
- Number of bytes
- Last modified time
- File/Directory name

`pwd`

```
 Will return current directory's name

Ex:
pwd

/home/manager
```

`cd`

```
 Change directory.
It is used for changing the working directory

Ex:
cd Documents
```

We are now in "Documents" directory. Command "cd .." takes you out from the current directory. Entering "cd" command will bring you to home directory.

`mkdir`

```
 make directory
This command creates a directory if no file/directory exists with that name.

Ex:
mkdir Practice
ls -l

drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
```

```
drwxrwxr-x 10 training training   4096 Jan 29 16:37 Data-Sreenu
drwxrwxr-x  3 training training   4096 Jan 29 14:22 bin
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
drwxrwxr-x  7 training training   4096 Jan 25 13:35 glue_dependencies
```

rmdir

```
 remove directory
This command removed an empty directory.

Ex:
rmdir Practice
ls -l

drwxrwxr-x 10 training training   4096 Jan 29 16:37 Data-Sreenu
drwxrwxr-x  3 training training   4096 Jan 29 14:22 bin
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
drwxrwxr-x  7 training training   4096 Jan 25 13:35 glue_dependencies
drwxr-xr-x  3 training training   4096 Jan 21 09:53 Downloads
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
drwxr-xr-x  7 training training   4096 Aug  3  2017 gluetools
```

touch

```
 It is file's timestamp changing command. However, it can be used for
creating an empty file. This command is generally used for checking
write permission for the user.

Ex:
touch temp-file
ls -l

drwxrwxr-x 31 training training   4096 Dec 19 13:15 anaconda2
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
drwxr-xr-x  7 training training   4096 Aug  3  2017 gluetools
drwxrwxr-x  3 training training   4096 Jan  9 09:58 Library
lrwxrwxrwx  1 training training     37 Jan  7 17:19 phyloscanner
drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
-rw-rw-r--  1 training training      0 Jan 30 14:01 temp-file
drwxrwxr-x 10 training training   4096 Jan 29 16:37 Data-Sreenu
```

rm

```
 remove
rm is used for removing files and directories.

Ex:
rm temp-file
ls -l

drwxrwxr-x 31 training training   4096 Dec 19 13:15 anaconda2
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
-rw-rw-r--  1 training training 669825 Jan 18 13:02 bamtools-master.zip
drwxrwxr-x  3 training training   4096 Jan 29 14:22 bin
drwxr-xr-x  2 training training   4096 Dec 18 20:19 Desktop
drwxrwxr-x  3 training training   4096 Jan  9 09:58 Library
lrwxrwxrwx  1 training training     37 Jan  7 17:19 phyloscanner
drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
drwxrwxr-x 11 training training   4096 Jan 30 15:23 Data-Sreenu
```

to remove directories use “-r” option. Please remember once a file or directory is deleted, it will not go to “Recycle bin” in Linux and there is no way you can recover it.

`cp`

```
 copy files or directories

Ex:
touch temp1
cp temp1 temp2
ls -l

drwxrwxr-x 31 training training   4096 Dec 19 13:15 anaconda2
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
-rw-rw-r--  1 training training 669825 Jan 18 13:02 bamtools-master.zip
drwxrwxr-x  3 training training   4096 Jan 29 14:22 bin
drwxr-xr-x  7 training training   4096 Aug  3  2017 gluetools
drwxrwxr-x  3 training training   4096 Jan  9 09:58 Library
lrwxrwxrwx  1 training training     37 Jan  7 17:19 phyloscanner
drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
-rw-rw-r--  1 training training      0 Jan 30 15:34 temp1
-rw-rw-r--  1 training training      0 Jan 30 15:34 temp2
drwxrwxr-x 11 training training   4096 Jan 30 15:23 Data-Sreenu
```

To copy directories, use “-r” option

mv

```
 to move file or directory
to rename

Ex:
mv temp1 Desktop/.
mv temp2 NewfileName
ls -l

drwxrwxr-x 31 training training   4096 Dec 19 13:15 anaconda2
drwxrwxr-x  5 training training   4096 Jan 18 14:42 bamtools-master
-rw-rw-r--  1 training training 669825 Jan 18 13:02 bamtools-master.zip
drwxrwxr-x  3 training training   4096 Jan  9 09:58 Library
-rw-rw-r--  1 training training      0 Jan 30 15:34 NewfileName
lrwxrwxrwx  1 training training     37 Jan  7 17:19 phyloscanner
drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
drwxrwxr-x 11 training training   4096 Jan 30 15:23 Data-Sreenu
```

In the first example “temp1” is move to Desktop directory. The “.” at the end will retain the file name (name is unchanged). In the second example temp2 is renamed to NewfileName.

ln

```
 link
to make links to files/directories.

Ex:
ln -s ~/Data-Sreenu/exFiles/Ebola-1.fa .
ls -l

drwxr-xr-x  2 training training   4096 Dec 18 20:19 Documents
drwxr-xr-x  3 training training   4096 Jan 21 09:53 Downloads
lrwxrwxrwx  1 training training     24 Jan 31 10:17 Ebola-1.fa -> ~/Data-Sreenu/exFiles/Ebola-1.fa
drwxrwxr-x  7 training training   4096 Jan 25 13:35 glue_dependencies
drwxr-xr-x  7 training training   4096 Aug  3  2017 gluetools
drwxrwxr-x  3 training training   4096 Jan  9 09:58 Library
-rw-rw-r--  1 training training      0 Jan 30 15:34 NewfileName
lrwxrwxrwx  1 training training     37 Jan  7 17:19 phyloscanner
drwxrwxr-x  2 training training   4096 Jan 30 13:46 Practice
drwxrwxr-x 13 training training   4096 Jan 29 14:22 Programs
drwxrwxr-x 10 training training   4096 Jan 30 15:55 Data-Sreenu
```

We encourage you to create links instead of copying data into various directories to save space.

1.5 File viewers

`cat`

```
 concatenate
combines files and prints on the screen

Ex:
cat ~/Data-Sreenu/exFiles/Ebola-1.fa

>gi|10313991|ref|NC_002549.1| Zaire ebolavirus isolate
CGGACACACAAAAAGAAAGAAGAATTTTTAGGATCTTTTGTGTGCGAATAACTATGAGGAAGATTAATAA
TTTTCCTCTCATTGAAATTTATATCGGAATTTAAATTGAAATTGTTACTGTAATCACACCTGGTTTGTTT
CAGAGCCACATCACAAAGATAGAGAACAACCTAGGTCTCCGAAGGGAGCAAGGGCATCAGTGTGCTCAGT
TGAAAATCCCTTGTCAACACCTAGGTCTTATCACATCACAAGTTCCACCTCAGACTCTGCAGGGTGATCC
AACAACCTTAATAGAAACATTATTGTTAAAGGACAGCATTAGTTCACAGTCAAACAAGCAAGATTGAGAA
TTAACCTTGGTTTTGAACTTGAACACTTAGGGGATTGAAGATTCAACAACCCTAAAGCTTGGGGTAAAAC
ATTGGAAATAGTTAAAAGACAAATTGCTCGGAATCACAAAATTCCGAGTATGGATTCTCGTCCTCAGAAA
ATCTGGATGGCGCCGAGTCTCACTGAATCTGACATGGATTACCACAAGATCTTGACAGCAGGTCTGTCCG
TTCAACAGGGGATTGTTCGGCAAAGAGTCATCCCAGTGTATCAAGTAAACAATCTTGAAGAAATTTGCCA
.............
```

`more/less`

```
 these commands are used for viewing the files.

Ex:
more ~/Data-Sreenu/exFiles/Ebola-1.fa

>gi|10313991|ref|NC_002549.1| Zaire ebolavirus isolate
CGGACACACAAAAAGAAAGAAGAATTTTTAGGATCTTTTGTGTGCGAATAACTATGAGGAAGATTAATAA
TTTTCCTCTCATTGAAATTTATATCGGAATTTAAATTGAAATTGTTACTGTAATCACACCTGGTTTGTTT
CAGAGCCACATCACAAAGATAGAGAACAACCTAGGTCTCCGAAGGGAGCAAGGGCATCAGTGTGCTCAGT
TGAAAATCCCTTGTCAACACCTAGGTCTTATCACATCACAAGTTCCACCTCAGACTCTGCAGGGTGATCC
AACAACCTTAATAGAAACATTATTGTTAAAGGACAGCATTAGTTCACAGTCAAACAAGCAAGATTGAGAA
TTAACCTTGGTTTTGAACTTGAACACTTAGGGGATTGAAGATTCAACAACCCTAAAGCTTGGGGTAAAAC
ATTGGAAATAGTTAAAAGACAAATTGCTCGGAATCACAAAATTCCGAGTATGGATTCTCGTCCTCAGAAA
ATCTGGATGGCGCCGAGTCTCACTGAATCTGACATGGATTACCACAAGATCTTGACAGCAGGTCTGTCCG
TTCAACAGGGGATTGTTCGGCAAAGAGTCATCCCAGTGTATCAAGTAAACAATCTTGAAGAAATTTGCCA
.............
```

Press “q” to come out from the program

`head/tail`

```
 these commands show first and last 10 lines respectively from a file

Ex:
head ~/Data-Sreenu/exFiles/Ebola-1.fa

>gi|10313991|ref|NC_002549.1| Zaire ebolavirus isolate
CGGACACACAAAAAGAAAGAAGAATTTTTAGGATCTTTTGTGTGCGAATAACTATGAGGAAGATTAATAA
TTTTCCTCTCATTGAAATTTATATCGGAATTTAAATTGAAATTGTTACTGTAATCACACCTGGTTTGTTT
CAGAGCCACATCACAAAGATAGAGAACAACCTAGGTCTCCGAAGGGAGCAAGGGCATCAGTGTGCTCAGT
TGAAAATCCCTTGTCAACACCTAGGTCTTATCACATCACAAGTTCCACCTCAGACTCTGCAGGGTGATCC
AACAACCTTAATAGAAACATTATTGTTAAAGGACAGCATTAGTTCACAGTCAAACAAGCAAGATTGAGAA
TTAACCTTGGTTTTGAACTTGAACACTTAGGGGATTGAAGATTCAACAACCCTAAAGCTTGGGGTAAAAC
ATTGGAAATAGTTAAAAGACAAATTGCTCGGAATCACAAAATTCCGAGTATGGATTCTCGTCCTCAGAAA
ATCTGGATGGCGCCGAGTCTCACTGAATCTGACATGGATTACCACAAGATCTTGACAGCAGGTCTGTCCG
TTCAACAGGGGATTGTTCGGCAAAGAGTCATCCCAGTGTATCAAGTAAACAATCTTGAAGAAATTTGCCA
.............
```

1.6 File editors

File viewers show the content of the file without making any changes. To change the file content you have to use file editors. There are many non-graphical text editors like ed, emacs, vi and nano are available on most of the Linux distributions. Some of them are very sophisticated (ex. vi) and for advanced users. Here we will be learning about a non-graphical file editor, "nano".

Nano (earlier called pico) is very much like any graphical editor without a mouse. All commands are executed through the use of the keyboard, using the <CTRL> key modifier. It can be used to edit virtually any kind of text file from the command line

Nano without a filename gives you a standard (blank) nano window.

At the bottom of the screen, there are commands with a symbol in front. The symbol tells that you need to hold down the Control (Ctrl) key, and then press the corresponding letter of the command you wish to use.

For Example:

Ctrl+X will exit nano and return you to the command line.

Nano Quick Reference

- Ctrl+X: Exit the editor. If you've edited text without saving, you'll be prompted as to whether you really want to exit.

- Ctrl+O: Write (output) the current contents of the text buffer to a file. A filename prompt will appear; press Ctrl+T to open the file navigator shown above.
- Ctrl+R: Read a text file into the current editing session. At the filename prompt, hit Ctrl+T: for the file navigator.
- Ctrl+K: Cut a line into the clipboard. You can press this repeatedly to copy multiple lines, which are then stored as one chunk.
- Ctrl+J: Justify (fill out) a paragraph of text. By default, this reflows text to match the width of the editing window.
- Ctrl+U: Uncut text, or rather, paste it from the clipboard. Note that after a Justify operation, this turns into unjustify.
- Ctrl+T: Check spelling.
- Ctrl+W: Find a word or phrase. At the prompt, use the cursor keys to go through previous search terms, or hit Ctrl+R to move into replace mode. Alternatively you can hit Ctrl+T to go to a specific line.
- Ctrl+C: Show current line number and file information.
- Ctrl+G: Get help; this provides information on navigating through files and common keyboard commands

Getting help in Linux
Most of the Linux commands have manual pages. To access them, use "man" or "info" command. The manual page gives a detailed explanation of the command, all available options and sometimes, also provides examples. For example to get the manual page for ls command type "man ls"

```
 LS(1)              User Commands                           LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about  the  FILEs (the current directory by default).
       Sort entries alphabetically if none of -cftuvSUX nor
       --sort is specified.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
```

```
              do not list implied . and ..

       --author
              with -l, print the author of each file
```

Please explore manual pages of all the above commands for available options.

1.7 Commands for text processing

`cut`

```
 The cut command is a command line utility to cut a section from a file.
Please see "man cut" for available option.

To cut a section of file use "-c" (characters)

Ex:
cut -c 1-10  ~/Data-Sreenu/exFiles/Ebola-1.fa

 >gi|103139
 CGGACACACA
 TTTTCCTCTC
 CAGAGCCACA
 TGAAAATCCC
 AACAACCTTA
 TTAACCTTGG
 ATTGGAAATA
 ATCTGGATGG
 TTCAACAGGG
```

The option "-c 1-10" will give you 1-10 characters from the input file.

Here are some of the useful options:

- -c: cut based on character position
- -d: cut based on delimiter
- -f: field number

We have a file named "viruses.txt" with all the virus names, genbank ids and genome length etc. These fields are separated by "|" symbol.

```
 head ~/Data-Sreenu/exFiles/viruses.txt
gi|167006425|ref|NC_010314.1| Abaca bunchy top virus DNA-N, complete genome|1090
gi|167006427|ref|NC_010315.1| Abaca bunchy top virus segment 2, complete sequence|1057
gi|167006428|ref|NC_010316.1| Abaca bunchy top virus DNA-S, complete genome|1087
gi|167006430|ref|NC_010317.1| Abaca bunchy top virus DNA-M, complete genome|1074
gi|167006432|ref|NC_010318.1| Abaca bunchy top virus DNA-C, complete sequence|1015
gi|167006434|ref|NC_010319.1| Abaca bunchy top virus DNA-R, complete genome|1099
```

To get only genbank id

```
 cut -d "|" -f2 ~/Data-Sreenu/exFiles/viruses.txt

167006425
167006427
167006428
167006430
167006432
```

Here "|" is a delimiter and second field is our choice of interest. Next try printing virus name (5th field) and it's length (6th field)

`sort`

It is used for sorting input content

Some options are:

- -t: field separator
- -n: numeric sort
- -k: sort with a key (field)
- -r: reverse sort
- -u: print unique entries

to sort viruses.txt based on their length.

```
sort -t "|" -nk6 ~/Data-Sreenu/exFiles/viruses.txt

gi|10518489|ref|NC_002566.1| Trichoplusia ni cytoplasmic, complete sequence|200
gi|18450262|ref|NC_003380.1| Rice yellow mottle virus satellite, complete genome|220
gi|21450051|ref|NC_004033.1| Turnip crinkle virus satellite RNA, complete genome|230
gi|9627203|ref|NC_001546.1| Arabis mosaic virus small satellite RNA  genome|300
gi|401715661|ref|NC_018451.1| Arabis mosaic virus small satellite genome|301
gi|14329174|ref|NC_002802.1| Discula destructiva virus 1 RNA 4, complete genome|308
gi|20087066|ref|NC_003533.1| Cereal yellow dwarf virus-RPV satellite RNA |322
```

`grep`

searches input for a given pattern

Some options are:

- -A: after context
- -B: before context
- -C: before and after context
- -c: count
- -l: file with match
- -i: ignore case
- -o: only match
- -v: invert match
- -w: word match

to get all Hepatitis viruses from virus.txt

```
grep Hepatitis ~/Data-Sreenu/exFiles/viruses.txt

gi|9629718|ref|NC_001837.1| Hepatitis GB virus A, complete genome|9550
gi|9628705|ref|NC_001710.1| GB virus C/Hepatitis G virus, complete genome|9392
gi|9626732|ref|NC_001489.1| Hepatitis A virus, complete genome|7478
gi|21326584|ref|NC_003977.1| Hepatitis B virus, complete genome|3215
```

Other text processing commands worth looking at are: tr, rev, sed, uniq, wc and paste.

1.8 I/O control in Linux

When you run a command the output will be sent to standard out (stdout) ie. terminal. However, we can send the stdout to a file using “>” redirection. This will redirect the standard output to a file.

```
 ls > list
cat list

anaconda2
bamtools-master
bamtools-master.zip
bin
Desktop
Documents
Downloads
```

This will create a new file called list with all the file names in the directory. Remember, if there is a file exists with a name “list”, its content will be overwritten by the stdout. Instead, we can append to a file using “>>” redirection.

Another kind of output that is generated by programs is standard error. We have to use “2>” to redirect it.

```
 ls /foo 2> error
```

To redirect stdout and stderr to a file use “&>”

1.9 Pipes

Piping in Linux is very powerful and efficient way to combine commands. Pipes (|) in Linux act as connecting links between commands. Pipe makes a previous commands output as next commands input. We can nest as many commands as we want using pipes. They play an important role in smooth running of the command flow and reducing the execution time.

To print 10 smallest viruses

```
 sort -t "|" -nk6 ~/Data-Sreenu/exFiles/viruses.txt  |head

gi|10518489|ref|NC_002566.1| Trichoplusia ni cytoplasmic polyhedrosis |200
gi|18450262|ref|NC_003380.1| Rice yellow mottle virus satellite, complete genome|220
```

```
gi|21450051|ref|NC_004033.1| Turnip crinkle virus satellite RNA, complete genome|230
gi|9627203|ref|NC_001546.1| Arabis mosaic virus small satellite RNA|300
gi|401715661|ref|NC_018451.1| Arabis mosaic virus small satellite complete genome|301
gi|14329174|ref|NC_002802.1| Discula destructiva virus 1 RNA 4, complete genome|308
gi|20087066|ref|NC_003533.1| Cereal yellow dwarf virus-RPV satellite RNA genome|322
gi|20522156|ref|NC_003798.1| Lucerne transient streak virus satellite RNA genome|324
gi|11119632|ref|NC_002602.2| Cucumber mosaic virus satellite RNA, complete genome|336
gi|55831391|ref|NC_006451.1| Turnip crinkle virus virulent satellite RNA C genome|355
```

We will be working on other examples during the course, where we use pipes to combine more than two commands.

1.10 Process control

Some commands take time to complete the job. For example if you want to compress a huge file with gzip command it might take few minutes to finish the job. Till it finishes you cannot run any command. In such situations, it is better to run this command in background by appending with "&" (we can also suspend a running job by Ctr-z and type bg).

```
 gzip Data-Sreenu/Sreenu/temp-file &
```

Once it completes the task we get a message saying "Done" . We can get list of currently jobs in the terminal by "jobs" command. This will give you all the background jobs running in the current terminal. If you want to see all the running processes in the system, use "top". You can get user specific details in top using "-u" option.

```
top -u $USER

Tasks: 177 total,   1 running, 176 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.3 us,  0.0 sy,  0.0 ni, 99.6 id,  0.1 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  8173000 total,  3493840 free,   631640 used,  4047520 buff/cache
KiB Swap:  8386556 total,  8385224 free,     1332 used.  7152328 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 5770 training  20   0  292648   5704   4608 S   0.0  0.1   0:00.01 gvfs-udisks2-vo
 5780 training  20   0  264652   3552   3180 S   0.0  0.0   0:00.00 gvfs-goa-volume
 5784 training  20   0  266596   3404   2960 S   0.0  0.0   0:00.00 gvfs-mtp-volume
 5788 training  20   0  411104   9492   6416 S   0.0  0.1   0:00.00 gvfs-afc-volume
 5793 training  20   0  278796   4024   3496 S   0.0  0.0   0:00.00 gvfs-gphoto2-vo
 6414 training  20   0   96076   4844   3452 S   0.0  0.1   0:00.09 sshd
 6415 training  20   0   29860   5468   3388 S   0.0  0.1   0:00.24 bash
 7250 training  20   0   49124   3824   3136 R   0.0  0.0   0:00.00 top
 8670 training  20   0  183584   5424   5424 S   0.0  0.1   0:33.31 dirmngr
 9605 training  20   0  173820   2112   2020 S   0.0  0.0   0:27.25 gpg-agent
13141 training  20   0   68492   4076   3616 S   0.0  0.0   0:00.10 gconfd-2
13142 training  20   0  338012   3668   3212 S   0.0  0.0   0:00.00 at-spi-bus-laun
18980 training  20   0   62868   6544   5488 S   0.0  0.1   0:00.11 systemd
18985 training  20   0  171204   2324     36 S   0.0  0.0   0:00.00 (sd-pam)
18997 training  20   0   43212   4040   3492 S   0.0  0.0   0:00.21 dbus-daemon
19022 training  20   0  290600   6640   5944 S   0.0  0.1   0:00.01 gvfsd
19027 training  20   0  428872   9688   6784 S   0.0  0.1   0:00.00 gvfsd-fuse
19038 training  20   0  187556   4908   4296 S   0.0  0.1   0:00.00 dconf-servic
```

“top” command prints 12 column dynamic output. These columns are...

- PID: Process Id, this is a unique number used to identify the process.
- User : The username of whoever launched the process.
- PR : Priority: The priority of the process. Processes with higher priority will be favored by the kernel and given more CPU time than processes with lower priority. Oddly enough, the lower this value, the higher the actual priority; the highest priority on *nix is -20 and the lowest is 20.
- NI: nice value : nice is a way of setting your process’ priority.
- VIRT: Virtual Memory Size (KiB) : The total amount of virtual memory used by the process.
- RES: Resident Memory Size (KiB) : The non-swapped physical memory a task has used.
- SHR: Shared Memory Size (KiB) : The amount of shared memory available to a task, not all of which is typically resident. It simply reflects memory that could be potentially shared with other processes.

- S: Process Status : The status of the task which can be one of:
  - D = uninterruptible sleep
  - R = running
  - S = sleeping
  - T = traced or stopped
  - Z = zombie
- %CPU: CPU Usage : The percentage of your CPU that is being used by the process. By default, top displays this as a percentage of a single CPU. On multi-core systems, you can have percentages that are greater than 100%. For example, if 3 cores are at 60% use, top will show a CPU use of 180.
- %MEM: Memory Usage (RES) : A task's currently used share of available physical memory (RAM).
- TIME+: CPU Time, hundredths : Total CPU time the task has used since it started.
- COMMAND: Command Name or Command Line : To see the full command line that launched the process, start top with the -c flag

If you want to stop a running background job use "kill" command.

```
kill 1234
```

This will kill the job with process id 1234. As a user you can kill only your jobs. You do not have permission to run this command on other users' process ids.

Command line shortcuts

- Up/Down arrows: Previous commands
- !!: Reruns previous command
- Tab: Auto complete
- Tab+Tab: All available options
- Ctr+a: Move cursor to start of line
- Ctr+e: Move cursor to end of line
- Ctr+ : Alternates between terminals
- Ctr+l: Clear screen ((or Command+k on Mac)
- Ctr+c: Terminates the running program
- Ctr+z: Suspends the running program
- Ctr+w: Removes a previous word
- Ctr+d: Logout
- Ctr+d(in a command): Removes a character
- Ctr+u: Removes till the beginning