Introduction to Linux

Roman Cheplyaka

Generic commands, files, directories

What am I running?

```
ngsuser@ubuntu:~$ cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=16.04
DISTRIB_CODENAME=xenial
DISTRIB_DESCRIPTION="Ubuntu 16.04 LTS"

ngsuser@ubuntu:~$ ps -p $$
  PID TTY          TIME CMD
   21 pts/0    00:00:00 bash
```

What is this command?

```
ngsuser@ubuntu:~$ type cat
cat is /bin/cat

ngsuser@ubuntu:~$ man cat

CAT(1)                          User Commands                          CAT(1)
NAME
       cat - concatenate files and print on the standard output
SYNOPSIS
       cat [OPTION]... [FILE]...
DESCRIPTION
       Concatenate FILE(s) to standard output.
       With no FILE, or when FILE is -, read standard input.
```

Find out what these are:

1. `vim`
2. `cd`
3. `for`

Where am I?

```
ngsuser@ubuntu:~$ pwd
/home/ngsuser

ngsuser@ubuntu:~$ ls

ngsuser@ubuntu:~$ ls -a
.    .bash_history  .bashrc    .tmux.conf
..   .bash_logout   .profile

ngsuser@ubuntu:~$ ls -al
total 20
drwxr-xr-x 2 ngsuser ngsuser   96 Jul 28 10:11 .
drwxr-xr-x 3 root    root      21 Jul 28 10:05 ..
-rw------- 1 ngsuser ngsuser    5 Jul 28 10:11 .bash_history
-rw-r--r-- 1 ngsuser ngsuser  220 Jul 28 10:05 .bash_logout
-rw-r--r-- 1 ngsuser ngsuser 3771 Jul 28 10:05 .bashrc
-rw-r--r-- 1 ngsuser ngsuser  655 Jul 28 10:05 .profile
-rw-r--r-- 1 root    root    1805 Jul 28 10:05 .tmux.conf
```

The Linux file system

```
/
|-- bin
|-- boot
|-- dev
|-- etc
|-- home
|   `-- ngsuser
|-- lib
|-- lib64
|-- media
|-- mnt
|-- opt
|-- proc
|-- root
|-- sbin
|-- sys
|-- tmp
|-- usr
`-- var
```

Creating directories

```
ngsuser@ubuntu:~$ mkdir ngschool
ngsuser@ubuntu:~$ mkdir ngschool/day1
ngsuser@ubuntu:~$ mkdir ngschool/day1/lecture1
```

Or:

```
ngsuser@ubuntu:~$ mkdir -p ngschool/day1/lecture1
```

Change directories

```
ngsuser@ubuntu:~$ cd ngschool/day1
ngsuser@ubuntu:~/ngschool/day1$ ls -l
total 0
drwxrwxr-x 2 ngsuser ngsuser 6 Jul 28 13:04 lecture1
```

These are equivalent:

```
ngsuser@ubuntu:~/ngschool/day1$ mkdir ../day2
ngsuser@ubuntu:~/ngschool/day1$ mkdir ~/ngschool/day2
ngsuser@ubuntu:~/ngschool/day1$ mkdir /home/ngsuser/ngschool/day2
```

Shortcuts

Go to ...	Command
Home directory	`cd`
Home directory (alt.)	`cd ~`
Subdirectory under home	`cd ~/ngschool`
Previous directory	`cd -`
Go up one level	`cd ..`
Go up two levels	`cd ../..`

Moving files around

Action	Command
Copy a file	`cp file1 file2`
Copy a file to another directory	`cp file1 ~/ngschool/`
Copy a directory	`cp -r ~/ngschool ~/ngschool2`
Rename a file/directory	`mv file1 file2`
Move a file/directory somewhere	`mv file1 ~/ngschool/`

Running things as root

```
ngsuser@ubuntu:~$ whoami
ngsuser

ngsuser@ubuntu:~$ sudo whoami
root

ngsuser@ubuntu:~$ sudo -i
root@ubuntu:~#
```

Practical: working with FASTQ files

Downloading a file

```
$ wget http://ngschool.local/downloads/DRR004004.fastq.gz

2016-07-30 14:20:27 (7.75 KB/s) - ‘DRR004004.fastq.gz’ saved [15438]
```

Useful wget options:

- `-b`: download in background
- `-c`: continue an interrupted download
- `-i file.txt`: read URLs from a text file

Uncompressing a file

Uncompress:

```
$ gunzip DRR004004.fastq.gz
```

- Removes the original compressed file `DRR004004.fastq.gz`
- Creates an uncompressed file named `DRR004004.fastq`
- `16Kb` $\rightarrow$ `83Kb`
- Better to keep the file compressed

Compress an uncompressed file:

```
$ gzip DRR004004.fastq
```

Working with a compressed file

Some standard commands have analogues that work on gzipped files:

- `zcat`
- `zless`
- `zgrep`
- ... and a few others

To look at the compressed file:

```
$ zless DRR004004.fastq.gz
$ zcat DRR004004.fastq.gz | head
```

Task: count the reads

Every record starts with `@`, so let's count those:

```
$ zgrep -c '^@' DRR004004.fastq.gz
```

- `zgrep` searches for strings in compressed files
  - for uncompressed files, it's simply `grep`
- `-c` means count the occurrences
  - without `-c`, it would print the occurrences
- `^` means only match the beginning of the line
- quotes are to prevent shell interpreting special characters
  - are not necessary in this case (`^` and `@` are not special), but don't hurt

Task: count the reads

```
$ zgrep -c '^@' DRR004004.fastq.gz
```

gives the wrong number of reads. Why?

Exercise: find the lines that break our algorithm

Task: count the reads

Approach #2: rely on the fact that every read occupies 4 lines

1. Count the number of lines
2. Divide it by 4

Read the manpage for the `wc` command to learn how to count lines.

Task: count the reads

```
$ zcat DRR004004.fastq.gz | wc -l
472

$ echo $(( $(zcat DRR004004.fastq.gz | wc -l) / 4 ))
118
```

Task: extract read sequences

```
$ zcat DRR004004.fastq.gz | awk 'NR % 4 == 2 {print}'
```

or simply

```
$ zcat DRR004004.fastq.gz | awk 'NR % 4 == 2'
```

Task: write duplicate reads to the file `dups.txt`

Hint 1: use `uniq`.

Hint 2: you'll also need to use `sort`.

To redirect the output of a command to a file, do:

```
$ command > file.txt
```

Note that this overwrites the previous file contents!

Task: extract reads, replace ACG with ACT

```
$ zcat DRR004004.fastq.gz | awk 'NR % 4 == 2' | \
sed -e 's/ACG/ACT/g'
```

Writes to the standard output; use > to redirect to a file.

Task: find the GC content of all the reads

The GC content is defined as

$$\frac{N_G + N_C}{N} = \frac{N_G + N_C}{N_G + N_C + N_A + N_T}$$

1. Find $N_G + N_C$ and write it to a variable `N_GC`:

   ```
   $ N_GC=$(zcat DRR004004.fastq.gz | \
   awk 'NR % 4 == 2' | grep -o '[GC]' | wc -l)
   ```

   Note: quotes are not optional here!

2. To find N, replace the pattern `[GC]` with a dot (`.`).
   Write the result to a variable `N`.

3. Use the `bc -l` to compute `$N_GC / $N`.

Task: put each read into its own fastq file

High-level algorithm:

1. Read the fastq file line by line and append the line into the current file
2. Every 4 lines, change the name of the current file

Task: put each read into its own fastq file

Put the code into a file called `split.sh`:

```
#!/bin/bash
nline=0
zcat DRR004004.fastq.gz | while read line; do
  filename=$(printf read-%.3d.fastq $((nline / 4)))
  printf "%s\n" "$line" >> "$filename"
  nline=$((nline+1))
done
```

Make it executable:

```
$ chmod +x split.sh
```

Run it:

```
$ ./split.sh
```

Task: rename each file to its read identifier

E.g. `read-006.fastq` → `DRR004004.7.fastq`

Use a for loop to iterate over files:

```
for file in read-*.fastq; do
  ...
done
```

Inside the loop, use `head` and `grep` to extract the sequence name.