Inspecting text files

FASTA Format

There is no single “file extension” for FASTA files, but there are many, the most common and generic being “.fasta” or “.fa”. Sometimes more specific versions are “.faa” for protein files (aminoacidic), and “.fna” for nucleic acids.

Let's start listing all the files ending with .fna in our home:

```
find ~ -name "*.fna"
```

How many sequences in those files? Last time we counted them first selecting the lines containing '>':

```
grep  '>' ~/examples/phage/vir_cds_from_genomic.fna
```

We subsequently pass the output of **grep** to the **wc** command, using a *pipe*. We used this trick mainly to make use of the pipes, but grep has a switch (`-c`, for count) for this task:

```
# Counting sequences in a file:
grep  '>' ~/examples/phage/vir_cds_from_genomic.fna | wc -l

# Counting sequences in multiple files using wildcards:
grep -c  '>' ~/examples/phage/*.fna
```

FASTQ Format

The FASTQ format devotes 4 lines for each sequence, the last being an encoded version of the quality score for each nucleotide. There are some FASTQ files in a shared directory called `/homes/qib/shared/reads/`. Let's have a look:

```
ls -l /homes/qib/shared/
head -n 8 /homes/qib/shared/phage_reads.fastq
```

How many reads? We can count the lines and then divide by 4!

```
wc -l /homes/qib/shared/phage_reads.fastq
```

Or we can use a specific bioinformatic tool:

```
seqkit stats  /homes/qib/shared/phage*.fa*
```

GFF: Annotation

The GFF (General Feature Format) is used to store annotations. An alternative format, called GTF, is more focused on genes annotations while GFF is more generic. They are both TSV (tab separated values), that is they are table where the boundaries across cells are marked by a single tabulation.

The first lines optionally specify some metadata, and they are preceded by a #.

Let's see an example:

```
less -S ~/examples/phage/vir_genomic.gff

# If we want to remove the header lines:
grep -v '^#' ~/examples/phage/vir_genomic.gff | less -S

# If we want to increase the tabulation:
grep -v '^#' ~/examples/phage/vir_genomic.gff | less -S -x 15
```

If we want to extract all the lines with CDSs, and then lines containing the word *capsid*:

```
grep -w CDS ~/examples/phage/vir_genomic.gff

grep -w CDS ~/examples/phage/vir_genomic.gff | grep -i capsid
```

A useful command to extract some columns from a text file is `cut`:

```
cut -f 1,3-5 ~/examples/phage/vir_genomic.gff
```

Other TSV

GFF, GTF, but also SAM and VCF are examples of tabular text files. They all are *tab-separated values*. A smaller example will be easier to deal with:

```
# Try using relative path!
cat /homes/qib/shared/enrolled.tsv
```

If we want to sort by username, that is the third column of the file:

```
sort -k 3 /homes/qib/shared/enrolled.tsv
```

Sometimes we need to increase the space used by tabs to have a clearer view:

```
sort -k 3 /homes/qib/shared/enrolled.tsv | less -S -x 20
```

Extracting information

Suppose that we want to make a list of names and surnames. In the “enrolled.tsv” table this column is missing. But we see that the email address can be used to infer the missing data. What can we do?

1. Extract the column with the email addresses
2. Treat it as a “@ separated values”, that is splitting the string where the @ is
3. Replace the dot with a space

Until now, we can already perform the first two steps

```
cut -f 2 /homes/qib/shared/enrolled.tsv | cut -f 1 -d "@"
```

And we finally can introduce a new command, `tr` that can convert a character to a new one:

```
cut -f 2 /homes/qib/shared/enrolled.tsv | cut -f 1 -d "@" | tr '.' ' '
```

As you can see from `man tr`, the **tr** program will use two characters and replace the first with the second. Here how to turn a TSV into a CSV, using a special symbol (\t) that means “tab”[1)]:

```
cat /homes/qib/shared/enrolled.tsv | tr '\t' ','
```

To write a mail to all of you i would need a line of all the email addresses, separated by “;”. We can replace the “new line” (\n):

```
cut -f 2 /homes/qib/shared/enrolled.tsv | tr '\n' ';'
```

Prepare for next workshop

Create a text file in your home directory called `~/reads.fasta` where you should put some substring taken from `~/examples/phage/vir_genomic.fna`.

You can extract a substring of at least 20 chars from anywhere. You can add errors (i.e. change some letters), small deletions or small insertions...

Decompressing archives

In your home there should be a couple of archives, in two very popular formats: “zip” and “tar.gz”. They are in your `examples/archives/` directory.

Unzipping is done by:

```
unzip FILENAME
```

while for *tar* archives:

```
tar xvfz FILENAME
```

The “switches” here are:

- x, to eXtract
- v, for Verbose reporting (print files as they are extracted). Don't add it if you are not interested in the list
- f, extract from a File (sounds crazy)
- z, the *tar* archive is alzo compressed with *gz*. Don't add it if the archive is `.tar` and not `.tar.gz`

Further readings

- File permissions

[1)]

similarily, `\n` means new-line