

CORSO DI METODI MOLECOLARI E BIOINFORMATICA

LM Biologia Evoluzionistica, Università di Padova

Dr. Enrico Gaffo, Dr. Andrea Binatti

Prof. Stefania Bortoluzzi

Esercitazione 3

Padova, 3 novembre 2021

Obiettivi dell'esercitazione

- Imparare ad usare alcuni comandi della shell per maneggiare file di testo
- Conoscere i formati file FASTA e GTF

Nella lezione precedente

Comandi e nozioni utili per muoversi nel filesystem

`pwd` (print working directory)

`cd` (change directory)

`ls` (list files)

`.` (directory corrente)

`..` (directory di un livello superiore)

Percorsi assoluti (ad es: `/home/vdi/Scaricati/myfile.png`)

Percorsi relativi (ad es: `Scaricati/myfile.png`)

Preparazione dati per esercitazione

Scaricare nella propria home directory il file dell'esercitazione 3 dal sito http://compgen.bio.unipd.it/~stefania/Didattica/AA2021-2022/MMOL_BIOINFO_BE/esercitazione3.zip

- Aprire un terminale: di default si è posizionati nella propria home directory:

N.B.: il comando `cd` senza argomenti ci porta alla home directory dell'utente

Accertarsi che il file sia presente nella directory corrente e/o di essere nella directory giusta (suggerimento: `ls`, `pwd`)

- Decomprimere l'archivio contenente i file per l'esercitazione usando il comando `unzip`:

```
$ unzip esercitazione3.zip
```

- Entrare nella directory `esercitazione4` con il comando `cd` e il path relativo verso la cartella `esercitazione3`:

```
$ cd esercitazione3
```

Preparazione dati per esercitazione

- Vedere il contenuto della directory. Usiamo l'opzione “-l” del comando **ls** per avere un file per riga:

```
$ ls -l
```

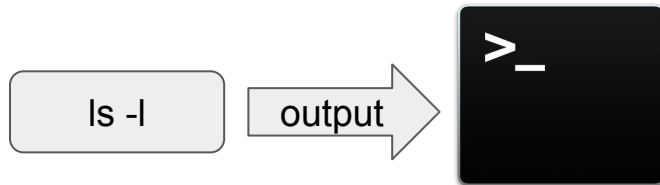
Verranno visualizzati **sul terminale** i nomi dei file presenti nella directory, tra cui 10 file in formato FASTA contenenti sequenze di trascritti riconducibili a geni coinvolti nel cancro al seno.

Preparazione dati per esercitazione

- Vedere il contenuto della directory. Usiamo l'opzione “-l” del comando **ls** per avere un file per riga:

```
$ ls -l
```

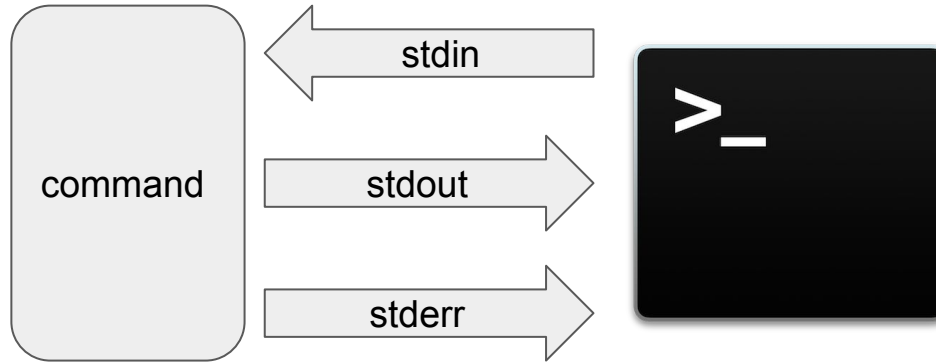
Verranno visualizzati **sul terminale** i nomi dei file presenti nella directory, tra cui 10 file in formato FASTA contenenti sequenze di trascritti riconducibili a geni coinvolti nel cancro al seno.



Flusso di dati nella shell: stdin, stdout, stderr

stdin, stdout, and stderr are three data streams created when you launch a Linux command

Error messages from the command are sent through the stderr (standard error) stream



Formato di file FASTA

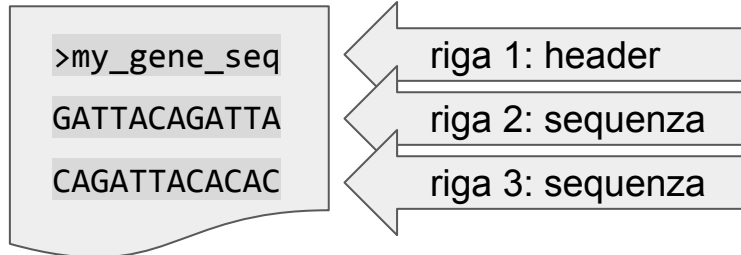
Il formato FASTA deriva dal programma di allineamento di sequenze di DNA e proteine “FASTA” (FAST-All). Esistono anche FASTP (proteins) e FASTN (nucleotides).

FASTA is a text-based format for representing either nucleotide sequences or amino acid (protein) sequences, in which nucleotides or amino acids are represented using single-letter codes

Un file in formato FASTA contiene una riga di intestazione che inizia con il carattere > seguito dal nome della sequenza (e.g. il nome del gene), le righe sottostanti l'intestazione sono la sequenza.

Solitamente hanno nomi del file con estensione .fasta o .fa

Un esempio minimale di una sequenza in formato FASTA:



Multi-FASTA file

Un file contenente molteplici sequenze in formato FASTA è detto multi-FASTA.

my_multifasta.fa

```
>gene1_seq  
GATTACAGATTA  
CAGATTACACAC
```

```
>gene2_seq  
CAGATTACA
```

```
>gene3_seq  
NACAGATTACA
```

Ogni sequenza FASTA ha la propria intestazione, quindi sequenze di entità (geni, trascritti, proteine, ...) diverse saranno riconoscibili.

Le sequenze genomiche sono disponibili nei database in file FASTA separati, uno per cromosoma, sia in un unico file in cui le sequenze dei cromosomi sono state concatenate.

Vedi <http://www.ensembl.org/info/data/ftp/index.html>

Visualizzare sul terminale (grandi) file di testo

Visualizzare il contenuto di uno dei file con il comando “less”:

```
$ man less
```

“Less does not have to read the entire input file before starting, so with large input files it starts up faster than text editors”

```
$ less seq01.fasta
```

Usare le frecce verso l’alto e verso il basso per scorrere il contenuto del file.

Per uscire dalla visualizzazione e tornare al prompt dei comandi premere il tasto **q** (“quit”).

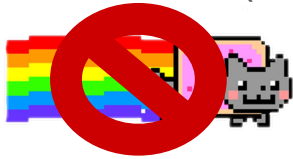
```
$ less -S seq01.fasta
```

-S or --chop-long-lines

Causes lines longer than the screen width to be chopped (truncated) rather than wrapped.

Creare un multi-FASTA: concatenare diversi file

Useremo il comando **cat** (concatenate) per concatenare due file FASTA (file di testo).



```
$ man cat
```

“cat [OPTION]... [FILE]...”

cat - concatenate files and print on the standard output”

```
$ cat seq01.fasta seq02.fasta
```

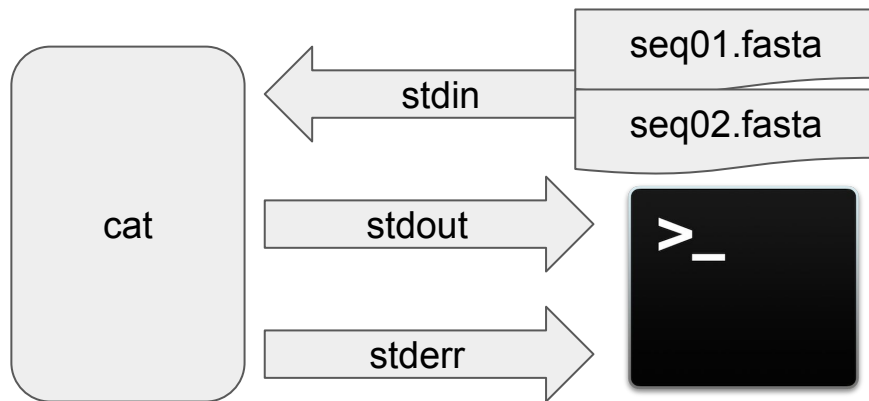
Creare un multi-FASTA: concatenare diversi file

“cat [OPTION]... [FILE]...”

cat - concatenate files and print on the standard output”

```
$ cat seq01.fasta seq02.fasta
```

Vedremo le due sequenze stampate nel terminale poichè lo stdout di cat è indirizzato al terminale



Creare un file multi-FASTA: l'operatore >

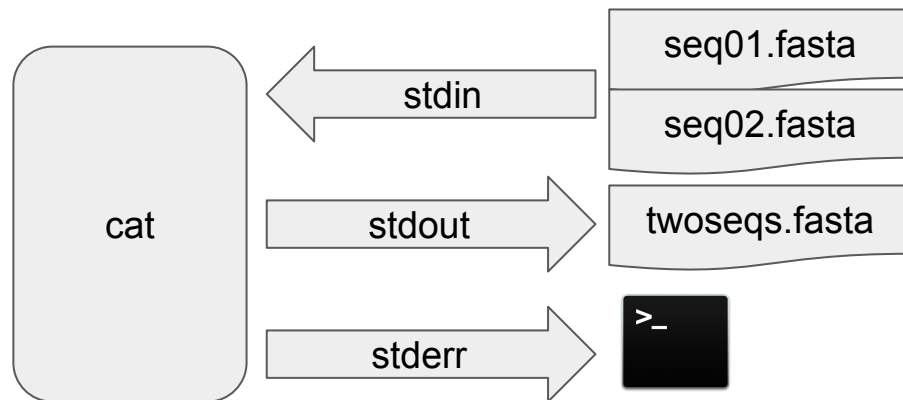
Per scrivere un nuovo file ci serviremo dell'operatore di redirectione dello standard output ">"

```
$ cat seq01.fasta seq02.fasta > twoseqs.fasta
```

">" crea un nuovo file o sovrascrive un file esistente!

N.B.: non confondere l'operatore di redirectione con il carattere indicatore di intestazione FASTA !

Il carattere ">" è il medesimo, ma i contesti sono diversi.



Creare un file multi-FASTA: concatenare file multipli

Creare un unico file FASTA a partire da singole sequenze, un file per sequenza: concateniamo tutte le sequenze FASTA in un unico file

```
$ cat seq01.fasta seq02.fasta [...] seq10.fasta > tenseqs.fasta
```

Una scorciatoia sfruttando le caratteristiche della shell:

usiamo un carattere “wild-card”, cioè un carattere speciale che sta ad indicare qualsiasi carattere (anche nessuno), ***** (l’asterisco)

```
$ cat seq*.fasta > all.fasta
```

seq*.fasta viene “espanso” ai nomi dei file che iniziano con “seq”, contengono 0 o più caratteri e poi finiscono con “.fasta”

(nelle esercitazioni successive capiremo meglio questo comportamento, quando vedremo le “espressioni regolari”)

Il mio primo multi-FASTA

Verifichiamo che il file appena creato contenga le sequenze dei vari file

```
$ less all.fasta
```

Abbiamo creato un file multi-FASTA !



Appendere file

E se ci fossimo dimenticati di aggiungere un file alla nostra catena?

L'operatore ">>" serve ad appendere l'output del comando a sinistra alla fine del file a destra

```
$ cat coli_genes.fasta >> all.fasta
```

Verifichiamo l'output

```
$ less all.fasta
```

Premendo "G" maiuscolo (Shift+g) less ci porta alla fine del file.

Per tornare all'inizio "g" (minuscolo)

Ottenere l'inizio e la fine di un file

Per visualizzare le prime N righe di un file

```
$ head all.fasta
```

Si può scegliere quante righe

```
$ head -n 20 all.fasta
```

```
( $ head -20 all.fasta )
```

Per visualizzare le ultime N righe di un file

```
$tail all.fasta
```

anche per tail si possono scegliere quante righe (contando a partire dal fondo)

```
$tail -n 22 all.fasta
```



Ottenere statistiche di un file di testo

Contare il numero totale di righe presenti nel file all.fasta: utilizzare il comando wc (word count !!)

```
$ wc all.fasta
```

```
516 536 30936 all.fasta
```

```
$ man wc
```

*“wc - print **newline**, **word**, and **byte** counts for each file”*



Alcune opzioni interessanti:

-l (lines)

```
$ wc -l all.fasta
```

```
516 all.fasta
```

-w (words)

```
$ wc -w all.fasta
```

```
536 all.fasta
```

-c (bytes)

```
$ wc -c all.fasta
```

```
30936 all.fasta
```

-m (characters)

```
$ wc -m all.fasta
```

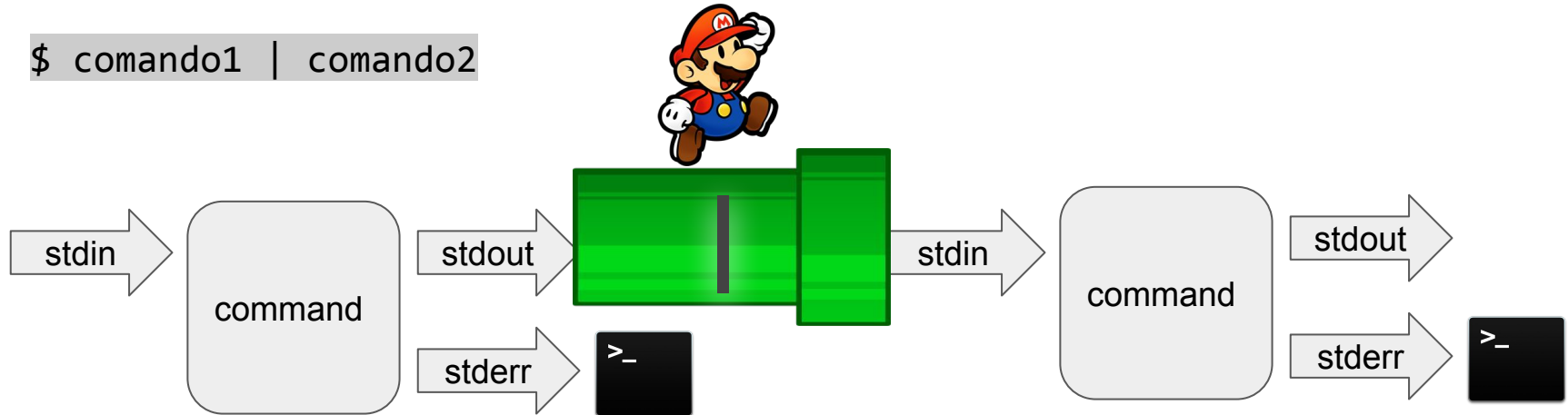
... è simile al numero di bytes?

L'operatore "pipe"

Con la shell Linux è possibile passare l'output di un comando direttamente ad un altro comando senza dover scrivere un file intermedio.

L'operatore | (pipe o barra verticale) collega l'output del comando a sinistra con l'input del comando alla sua destra:

```
$ comando1 | comando2
```



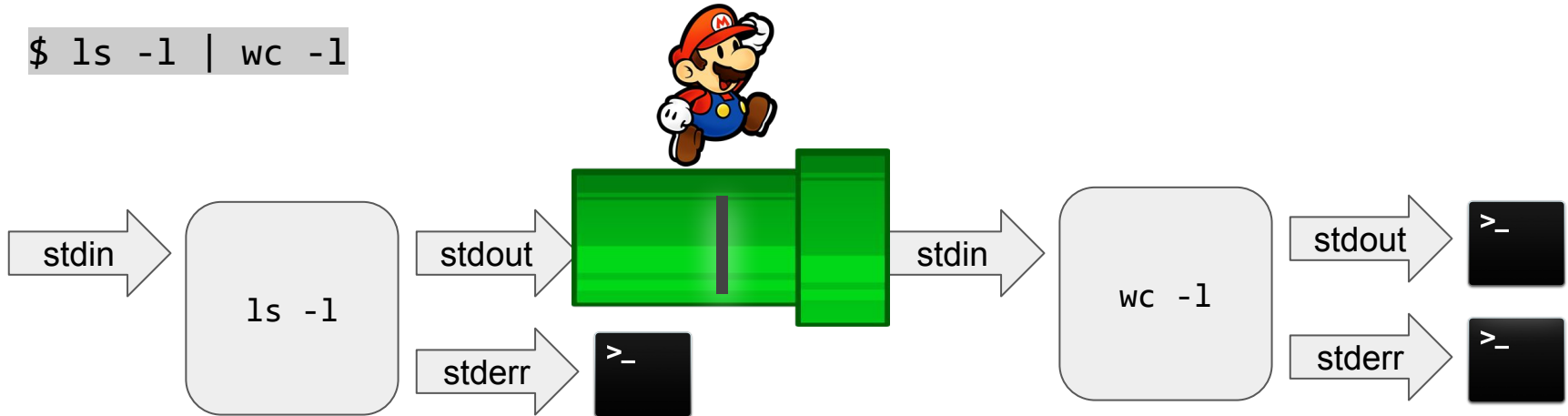
L'operatore "pipe"

Ad esempio, un altro modo per contare il numero di righe nel file all.fasta è:

```
$ cat all.fasta | wc -l
```

Oppure, contare quanti file ci sono nella directory

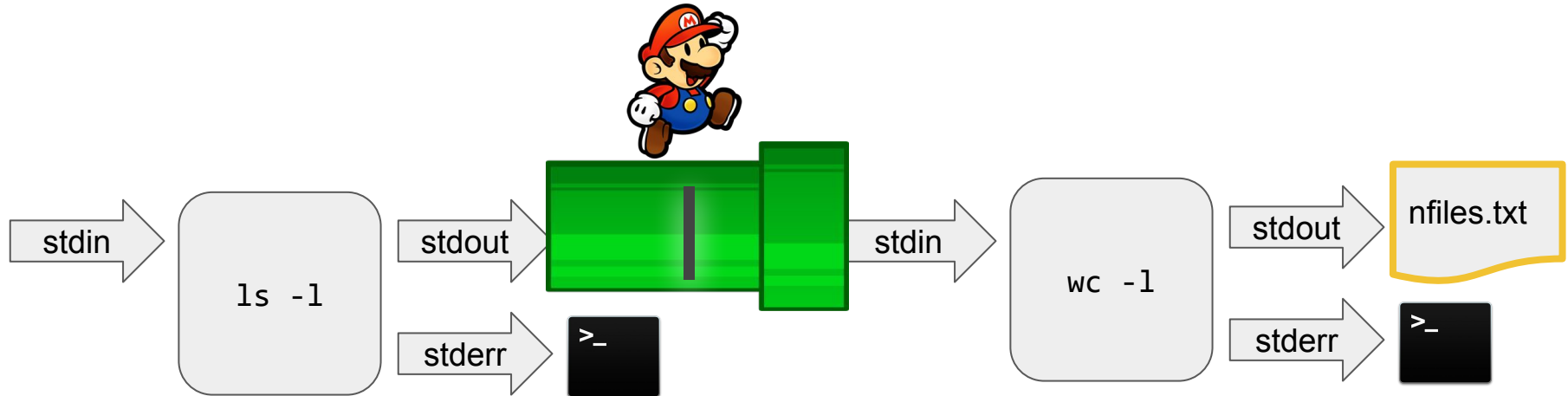
```
$ ls -l | wc -l
```



Una semplice pipeline di comandi

Contare quanti file ci sono nella directory e salvare il risultato in un file

```
$ ls -l | wc -l > nfiles.txt
```



Verificare il numero di sequenze nel multi-FASTA creato

Usare il comando **grep** che permette di filtrare le righe di un file che hanno un match con un dato pattern:

```
$ grep ">" all.fasta
```

N.B.: è importante mettere le virgolette intorno al maggiore dato che è un carattere che ha un significato preciso nella shell (redirezionamento dell'output di un comando).

Contare quante righe vengono date in output:

```
$ grep ">" all.fasta | wc -l
```

Sei in panico perché il terminale non risponde ai comandi?

CTRL + c

combinazione di comandi per interrompere il processo nella console e riottenere il prompt del terminale

Contare il numero di nucleotidi in un file (multi-)FASTA

Se vogliamo ottenere informazioni sul comando grep o su qualsiasi altro comando, possiamo usare il comando man:

```
$ man grep
```

oppure usiamo l'opzione del comando che mostra l'help:

```
$ grep --help
```

Per ottenere il numero approssimativo di nucleotidi totali presenti nelle nostre 10 sequenze, utilizziamo l'opzione -v di grep che indica a grep di invertire il match, cioè di NON mostrarci le righe che hanno il match:

```
$ grep -v ">" all.fasta | wc -c
```

Da notare che il numero ottenuto di caratteri che otteniamo non è preciso dato che vengono conteggiati anche i new line, un carattere “invisibile” che viene interpretato come “andare a capo su una nuova riga”.

Il formato file GTF per le annotazioni geniche

Il formato GTF (General Transfer Format) è un file di testo usato per raccogliere le annotazioni di una regione genomica. Consiste di una linea per feature (gene, trascritto, esone), ognuna formata da 9 colonne di dati.

Per una descrizione dettagliata del formato GTF consultare la seguente pagina web di Ensembl:

<http://m.ensembl.org/info/website/upload/gff.html>

Dare un'occhiata al file con il comando less:

```
$ less sample.gtf
```

Contiamo il numero di righe nel file:

```
$ wc -l sample.gtf
```

Ottenere porzioni di file (~ colonne)

Vogliamo ora vedere quanti geni sono stati identificati nel cromosoma V.

Ci serve il comando cut: “*cut out selected portions of each line of a file*”.

Quindi cut ci permette di selezionare soltanto alcune porzioni da ogni riga del file. In particolare, le porzioni (colonne) vengono selezionate con l'opzione -f e devono essere delimitate da un qualche carattere detto delimitatore (nel nostro caso è il TAB).

Facciamo in modo di stampare il primo campo (il numero del cromosoma):

```
$ cut -f 1 sample.gtf | less
```

Filtrare delle righe in base al loro contenuto

Ora selezioniamo solo le righe che contengono il cromosoma V:

```
$ cut -f 1 sample.gtf | grep -w "^V" | less
```

“^V” è una regular expression che specifica tutte le parole che sono all’inizio della riga (^) e che iniziano con V

Contiamo quante sono:

```
$ cut -f 1 sample.gtf | grep -w "^V" | wc -l
```

Infine, proviamo a fare la stessa cosa partendo dal file GTF contenente anche l’intestazione, senza creare un file intermedio:

```
$ grep -v "^#" sample.gtf | cut -f 1 | grep -w "^V" | wc -l
```

Filtrare delle righe in base al loro contenuto

Questo comando ci mostra il numero totale di righe nel file compresa l'intestazione.

A noi, invece, interessa sapere quanti geni sono presenti nel file, quindi procederemo con il creare un file GTF senza l'intestazione:

```
$ grep -v "^#" sample.gtf > sample_no_header.gtf
```

-v indica a grep di invertire il match, quindi di NON mostrare le righe che “matchano” la regular expression

^# è una regular expression che specifica tutte le parole che sono all'inizio della riga (^) e che iniziano con # → questa è una caratteristica unica delle righe presenti nell'intestazione del file GTF.

Contiamo ora quanti geni ci sono all'interno del nostro file:

```
$ wc -l sample.gtf
```