

Programación con Shell Scripting: Sesión 5

Máster Universitario en Bioinformática



Universidad
Internacional
de Valencia

Dra. Paula Soler Vila
paula.solerv@professor.universidadviu.com

De:
 Planeta Formación y Universidades

Aspectos a tratar

1

Comandos para el procesamiento básico de archivos

- *Cat*
- *Paste*
- *Join*
- *Diff*
- *Cut*
- *Sort*
- *Uniq*
- *Tr*
- *Wc*
- *Rev*
- *Fold*

2

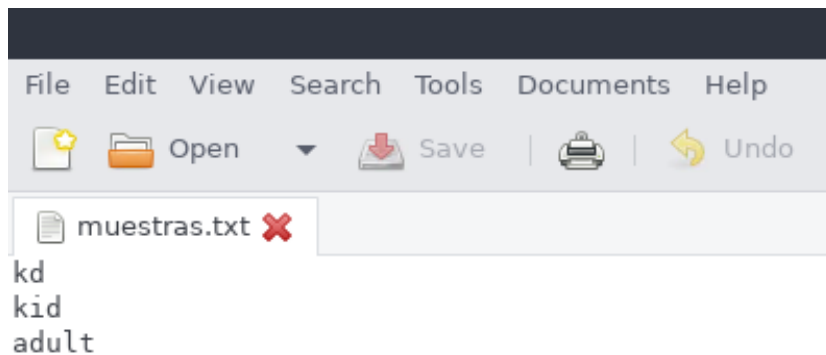
Ejercicio con **GENCODE**: procesamiento básico de datos

¿Puedo modificar el contenido de mi archivo con el comando **cat**?

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat > muestras.txt  
kd  
kid  
adult
```

Uso de editores de texto

pluma muestras.txt



Uso de otros comandos

```
$ sed 's/kd/kid/' muestras.txt  
kid  
kid  
adult  
  
$ sed -i 's/kd/kid/' muestras.txt  
$ cat muestras.txt  
kid  
kid  
adult
```

¿Qué tienen en común comandos tales como *diff* o *join*?



ORDENACIÓN PREVIA DE LOS DATOS

Comando **sort**

Permite **ordenar líneas** de archivos de entrada utilizando **ciertos criterios** de ordenamiento

```
sort <opciones> <archivo>
```

1. Verificar si los datos de entrada están ordenados (-c)
2. Ordenar alfabéticamente
3. Clasificar en orden inverso (-r)
4. Ordenar por número (-n) **hay que determinarle a sort con -n que trabajamos con números porque si no no los considera numeros**
5. Ordenar y eliminar duplicados (-u)
6. Ordenar por elementos que no están al principio de la línea (-k)

Comando *sort -> history*

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat > genes.txt
```

Esto para crear el archivo, y con pluma genes.txt lo puedes editar

```
pax4
```

```
sox13
```

```
sox13
```

```
baf1
```

```
tcf4
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -c genes.txt
```

```
sort: genes.txt:4: disorder: baf1
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort genes.txt
```

Si los datos están ordenados
no va a reportar ninguna
salida

Control d, salir y guardar?

```
baf1
```

```
pax4
```

```
sox13
```

```
sox13
```

```
tcf4
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -u genes.txt
```

```
baf1
```

```
pax4
```

```
sox13
```

```
Tcf4
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -u genes.txt -o genes_output.txt
```

para cambiar la salida del archivo

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat genes_output.txt
```

```
baf1
```

```
pax4
```

```
sox13
```

```
tcf4
```

Comando *sort + shuf -> history*

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -u genes.txt -o genes_output.txt
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat genes_output.txt
```

baf1

pax4

sox13

tcf4

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -r genes_output.txt
```

tcf4

sox13

pax4

baf1

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ shuf -i 1-10 -n 10 > numeros.txt
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ sort -n numeros.txt
```

1

2

3

4

5

6

7

8

9

10

shuf -i es para que indique números aleatorios por ejemplo del 1 al 10, **-n** para indicar cuantos números quieres que te de

Comando *sort + seq -> history*

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ seq 3
```

```
1
```

```
2
```

```
3
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ seq 2 5
```

```
2
```

```
3
```

```
4
```

```
5
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ seq 1 2 6
```

```
1
```

```
3
```

```
5
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ seq 5 -1 1
```

```
5
```

```
4
```

```
3
```

```
2
```

```
1
```

El número del medio indica el intervalo de la secuencia que queremos seguir

Comando *sort -> history*

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	3.9G	0	3.9G	0%	/dev
tmpfs	3.9G	8.0K	3.9G	1%	/dev/shm
tmpfs	3.9G	672K	3.9G	1%	/run
tmpfs	3.9G	0	3.9G	0%	/sys/fs/cgroup
/dev/nvme0n1p1	80G	14G	67G	18%	/

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ df -h | sort -n -k 4
```

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	3.9G	0	3.9G	0%	/dev
tmpfs	3.9G	0	3.9G	0%	/sys/fs/cgroup
tmpfs	3.9G	672K	3.9G	1%	/run
tmpfs	3.9G	8.0K	3.9G	1%	/dev/shm
/dev/nvme1n1p1	99G	82G	17G	84%	/volumes/user

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ df -h | sort -k 4 -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
tmpfs	784M	56K	784M	1%	/run/user/71085
devtmpfs	3.9G	0	3.9G	0%	/dev
tmpfs	3.9G	0	3.9G	0%	/sys/fs/cgroup
tmpfs	3.9G	672K	3.9G	1%	/run

Como se puede ver hay un valor en M que lo ha puesto mas grande que los GB, por tanto está mal ordenado, y es porque el comando es `-n` que **SOLO CONTEMPLA NÚMEROS** por tanto, para que también tenga en cuenta el caracter M o GB habrá que utilizar `-h`

Comando *uniq* (*unique*)

uniq se utiliza para informar u omitir cadenas o líneas repetidas

```
uniq <opciones> <archivo>
```

esto será importante en secuencias de DNA y de genes

1. Ver la cantidad de veces que se repite una línea (-c)
2. Imprimir en pantalla las líneas repetidas, y obviar las no repetidas (-d)
3. Imprimir en pantalla las líneas no repetidas (-u)
4. Ignorar mayúsculas y minúsculas (-i)

Comando *uniq (history)*

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat uniq.txt
```

```
prueba1  
prueba1  
prueba1  
Prueba1  
Prueba2  
prueba2  
test1  
test1  
Test1
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ uniq -c uniq.txt
```

```
3 prueba1  
1 Prueba1  
1 Prueba2  
1 prueba2  
2 test1  
1 Test1
```

con -c te está contando sin tener en cuenta las mayúsculas y minúsculas

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ uniq -u uniq.txt
```

```
Prueba1  
Prueba2  
prueba2  
Test1
```

con -u te dice las que son únicas, las que no están repetidas

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ uniq -d uniq.txt
```

```
prueba1  
test1
```

-d nos dice que opciones están repetidas

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ uniq uniq.txt
```

```
prueba1  
Prueba1  
Prueba2  
prueba2  
test1  
Test1
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ uniq -i uniq.txt
```

```
prueba1  
Prueba2  
test1
```

Comando *uniq*

¿Qué ocurre si **NO** está ordenado?

```
cat uniq2.txt
```

```
prueba1  
prueba1  
prueba1  
Prueba1  
Prueba2  
prueba2  
test1  
test1  
Test1  
prueba1
```



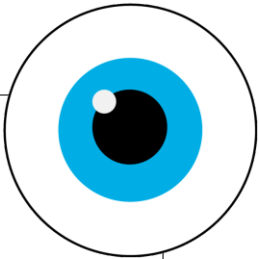
```
uniq uniq2.txt
```

```
prueba1  
Prueba1  
Prueba2  
prueba2  
test1  
Test1  
prueba1
```

IMP EXAMEN !!

**No detecta entradas
duplicadas que no se
encuentren en líneas
adyacentes
(comando *SORT*)**

**Es decir no va a encontrar lineas duplicadas que no
se encuentren una debajo de la otra!!**



Comando *tr (translate)*

Se usa para:

1. Cambiar caracteres:
 - Mayúsculas a minúsculas o viceversa.
 - Reemplazarlos por otros.
2. Borrar caracteres.

NO transforma palabras completas
Trabaja carácter a carácter

Sintaxis básica

```
tr <opciones> <conjunto_caracter_1> <conjunto_caracter_2>
```

Comando *tr (translate) -> history*

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ echo "Esto es un ejemplo de tr para la clase de hoy" | tr 'a' 'A'  
Esto es unA ejemplo de tr pArA lA clAsE de hoy
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ echo "Esto es un ejemplo de tr para la clase de hoy" | tr 'aeiou' 'AEIOU'  
EstO Es UnA EjEmpLO dE tr pArA lA clAsE dE hOy
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ cat /etc/passwd | tr "aeiou" "AEIOU" | head
```

```
rOOt:x:0:0:rOOt:/rOOt:/bIn/bAsh
```

```
bIn:x:1:1:bIn:/bIn:/sbIn/nOIogIn
```

```
dAEmOn:x:2:2:dAEmOn:/sbIn:/sbIn/nOIogIn
```

```
Adm:x:3:4:Adm:/vAr/Adm:/sbIn/nOIogIn
```

```
lp:x:4:7:lp:/vAr/spOOl/lpd:/sbIn/nOIogIn
```

```
sync:x:5:0:sync:/sbIn:/bIn/sync
```

```
shUtdOwn:x:6:0:shUtdOwn:/sbIn:/sbIn/shUtdOwn
```

```
hAlt:x:7:0:hAlt:/sbIn:/sbIn/hAlt
```

```
mAll:x:8:12:mAll:/vAr/spOOl/mAll:/sbIn/nOIogIn
```

```
OpErAtOr:x:11:0:OpErAtOr:/rOOt:/sbIn/nOIogIn
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr ~]$ tr "aeiou" "AEIOU" < /etc/passwd | head
```

Para cambiar espacios, se designan por “ ” vacío, por tabulador, se designa por “\t”

También se puede poner el espacio como [:space:], pero elimina el salto de línea

ESTE ES UN COMANDO que requiere de la REDIRECCIÓN DE LA SALIDA ESTÁNDAR para poder ser ejecutado

IMP PREGUNTA DE EXAMEN

el comando tr acepta el contenido de un fichero siempre y cuando utilicemos un comando como cat o a través de la redirección de la entrada estándar con <

Comando *tr (translate) -> history*

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "Esto es un ejemplo de tr para la clase de hoy" | tr 'a-z' 'A-Z'
```

ESTO ES UNA EJEMPLO DE TR PARA LA CLASE DE HOY **Para cambiar todas las letras de minúsculas a mayúsculas**

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "Esto es un ejemplo de tr para la clase de hoy" | tr -d "e"
```

Esto s un jmplo d tr para la clas d hoy

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "Esto es un ejemplo de tr para la clase de hoy" | tr -d "Ee"
```

sto s un jmplo d tr para la clas d hoy

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "AAA TTT GGG CCC"
```

AAA TTT GGG CCC

**con -c sustituye todos los
caracteres que no sean ATGC por
un guión**

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "AAA TTT GGG CCC" | tr -c "ATGC" "-"
```

AAA-TTT-GGG-CCC-(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "AAA TTT GGG CCC" | tr -c "ATGC\n" "-"
```

AAA-TTT-GGG-CCC

aquí le estamos indicando que haga salto de línea

Comando *wc* (*word count*)

```
wc <opciones> <archivo(s)>
```

Sin opciones : cuenta líneas, palabras y caracteres

Opciones

- c bytes
- m caracteres
- l líneas
- w palabras
- L longitud de la línea más larga

Comando *wc* (*word count*)

```
> cat sequences.txt
```

```
1      ACGT
```

```
2      ACGT
```

```
3      TTTGACA
```

```
> wc sequences.txt
```

¿Cuál será el resultado?

3 líneas

6 palabras

24 caracteres

las palabras son los caracteres separados por espacios en blanco

1	\t	ACGT \n
2	\t	ACGT \n
3	\t	TTTGACA\n

**Se cuenta los caracteres que se ven y
aquellos que no se ven**

se cuenta tanto los espacios \t como los saltos de línea \n

Comando *wc* (*word count*)

```
> cat sequences.txt
```

```
1      ACGT
2      ACGT
3      TTTGACA
```

Si le pones -n por ejemplo: `echo -n "Esto es una prueba"` no va a hacer el salto de línea, entonces cuando hagas `echo -n "Esto es una prueba" | wc`, ese salto de línea no lo va a contar, pero si no, por defecto sí lo cuenta

```
> wc sequences.txt
```

Con `wc -l`, va a contar las líneas que tienen salto de línea solamente. Sin embargo, si utilizas `cat -n`, va a contar todas las líneas que haya aunque no tenga salto de línea

¿Cuál será el resultado?

3 líneas

6 palabras

24 caracteres



1	\t	ACGT \n
2	\t	ACGT \n
3	\t	TTTGACA\n

PRACTIQUEMOS



Comando *rev*

rev se utiliza para invertir las líneas de texto en función de los *caracteres*

```
rev <opciones> <archivo>
```

PRACTIQUEMOS



Comando *rev (history)*

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ rev
```

```
paula  
aluap  
hola  
Aloh
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ echo "ATGC" | rev
```

```
CGTA
```

```
[UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ cat > sequences.txt
```

```
AAATTT  
GGGCC  
CATG
```

```
(base) [UNIVERSIDADVIU\paula.soler@a-3edhijmqygwxr~]$ rev sequences.txt
```

```
TTTAA  
CCGGG  
GTAC
```

Comando *fold*

Permite dividir las líneas en un ancho especificado

fold <opciones> <fichero>

```
$ cat fold1.txt
```

[illegible]

```
$ fold -w 80 fold1.txt
```

por defecto el valor es 80 y se recomienda que no supere ese tamaño

[illegible]

```
$ fold -w 30 fold1.txt
```

[illegible]

Comando *fold*

> **fold -w 30** fold1.txt

El computador puede ser definido como una máquina electrónica que realiza tareas de procesamiento, almacenamiento y movimiento de datos junto con el control de los procesos requeridos. Para ello, ejecuta una secuencia de instrucciones, conocidas como programas, que le permiten desarrollar la tarea asignada (Tanenbaum, 2016). A pesar de ser una colección de partes altamente sofisticadas, es simplemente eso, una colección de piezas muy sofisticadas.


-S

esto es para que no rompa
las palabras, y separe cuando
haya espacios


> **fold -w30 -s** fold1.txt

El computador puede ser definido como una máquina electrónica que realiza tareas de procesamiento, almacenamiento y movimiento de datos junto con el control de los procesos requeridos. Para ello, ejecuta una secuencia de instrucciones, conocidas como programas, que le permiten desarrollar la tarea asignada (Tanenbaum, 2016). A pesar de ser una colección de partes altamente sofisticadas, es simplemente eso, una colección de piezas muy sofisticadas.

Combinando lo aprendido



[Human](#)[Mouse](#)[How to access data](#)[FAQ](#)[Documentation](#)[About us](#)



Human

Release 45 (GRCh38.p14)

- [Statistics of this release](#)
- [More information about this assembly](#) (including patches, scaffolds and haplotypes)
- [Go to GRCh37 version of this release](#)

GTF / GFF3 files

Content	Regions	Description	Download
Comprehensive gene annotation	CHR	<ul style="list-style-type: none">It contains the comprehensive gene annotation on the reference chromosomes only	GTF GFF3
Comprehensive gene annotation	ALL	<ul style="list-style-type: none">It contains the comprehensive gene annotation on the reference chromosomes, scaffolds, assembly patches and alternate loci (haplotypes)	GTF GFF3
Comprehensive gene annotation	PRI	<ul style="list-style-type: none">It contains the comprehensive gene annotation on the primary assembly (chromosomes and scaffolds) sequence regions	GTF GFF3
Basic gene annotation s	CHR	<ul style="list-style-type: none">It contains the basic gene annotation on the reference chromosomes onlyThis is a subset of the corresponding comprehensive annotation, including only those transcripts tagged as 'basic' in every geneThis is the main annotation file for most users	GTF GFF3
Basic gene annotation	ALL	<ul style="list-style-type: none">It contains the basic gene annotation on the reference chromosomes, scaffolds, assembly patches and alternate loci (haplotypes)This is a subset of the corresponding comprehensive annotation, including only those transcripts tagged as 'basic' in every gene	GTF GFF3

More about GENCODE Human

- [Current human data](#)
- [Release history](#)
- [Statistics](#)
- [Data format](#)
- [FTP site](#)

<https://www.gencodegenes.org/human/>

Prueba aplicativa 1

Prueba aplicativa 1

Disponibilidad: El elemento está oculto para los estudiantes. Estará disponible después del 30-abr-2024 22:00.

Archivos adjuntos:

- human_coordinates_1.bed (45,321 KB)
- human_coordinates_2.bed (45,39 KB)
- selected_genes.txt (2,886 KB)
- Actividad 1.docx (229,304 KB)
- Actividad 1.pdf (534,258 KB)



Actividad 1.- Manipulación y formateo de archivos: Formato BED

El objetivo de esta actividad es que el estudiante adquiera habilidades en la manipulación y formateo de archivos utilizando comandos de Linux, que han sido aprendidos a lo largo de las sesiones teóricas de la asignatura. En particular, se enfocará en el formato BED (*Browser Extensible Data*) que se utiliza extensamente en bioinformática para almacenar regiones genómicas, como coordenadas y anotaciones asociadas. Este formato se caracteriza por presentar los datos en forma de columnas separadas por espacios o tabuladores.

Instrucciones de entrega

- La entrega se realizará a través del Campus VIU en un archivo único en formato PDF utilizando este documento como plantilla. Recuerde que las actividades a realizar están resaltadas en negrita.
- Incluya el código empleado, capturas de pantalla con su usuario (agregando el *prompt* completo) y resolución máxima.
- Proporcione explicaciones claras y concisas de los comandos utilizados. Si los comandos empleados no se explican brevemente, el valor de la pregunta será penalizado a la mitad.
- Reportar solo una opción/forma para resolver las distintas preguntas planteadas.



viu

Universidad
Internacional
de Valencia

universidadviu.com

De:
 Planeta Formación y Universidades