

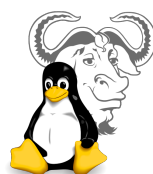
# Eines Informàtiques per a l'Estadística: Bash

Joaquim Roé i Albert Ruiz

Grau d'Estadística Aplicada

Universitat Autònoma de Barcelona

Versió: 28 d'octubre de 2021



## Índex

<b>1</b>	<b>Bash i el sistema operatiu</b>	<b>3</b>
1.1	Introducció al sistema operatiu GNU-Linux . . . . .	3
1.1.1	Iniciant una sessió en GNU-Linux als ordinadors de la facultat . . . . .	4
1.1.2	Iniciant una sessió en GNU-Linux remota per Cocalc . . . . .	5
1.1.3	Des d'una instal·lació pròpia o una màquina virtual . . . . .	5
1.1.4	Bash en un dispositiu mòbil . . . . .	6
1.2	Primers passos . . . . .	6
1.2.1	Navegació amb terminal . . . . .	7
1.3	Obtenció d'ajuda . . . . .	9
1.4	Edició de fitxers . . . . .	10
1.5	Compressió de fitxers . . . . .	11
<b>2</b>	<b>Manipulació de fitxers</b>	<b>11</b>
2.1	Sistema de directoris. Crear, moure i esborrar . . . . .	11
2.2	Entrada i sortida. Les "pipes" . . . . .	13
2.3	Ordenació de dades: <i>sort</i> . . . . .	14
2.4	Sistema de fitxers: permisos . . . . .	16

<b>3 Edició i manipulació de fitxers</b>	<b>17</b>
3.1 <i>grep</i> i expressions regulars . . . . .	17
3.1.1 Expressions de l' <i>awk</i> . . . . .	20
3.1.2 Instruccions de l' <i>awk</i> . . . . .	21
<b>Apèndix: La consola del Windows</b>	<b>22</b>

# Bash

## 1 Bash i el sistema operatiu

Tots els sistemes operatius incorporen un programa *intèrpret d'ordres*, també anomenat *emulador de terminal* o *consola*, que té la capacitat de comunicar ordres dels usuaris (introduïdes per teclat) al sistema operatiu i al conjunt de programes instal·lats en un ordinador, i presentar en forma de text en pantalla les respostes del sistema operatiu i altres programes. És una forma de treball interactiva en què usuari i màquina es comuniquen en forma successiva. En aquest bloc de l'assignatura aprendrem a fer servir el **Bash**, que és la consola utilitzada pels sistemes operatius que s'ajusten a l'estàndard POSIX, incloent el GNU-Linux, altres formes de Unix, MacOS, o Android.

Per comprendre les interaccions entre Bash i els altres components del sistema operatiu, utilitzarem essencialment tres programes:

- **Bash**. Si treballem en un ordinador amb Linux, MacOS, o un altre sistema POSIX, aquest és l'intèrpret d'ordres per defecte i no necessitarem instal·lar res més. Si treballem en un ordinador amb Microsoft Windows, utilitzarem el Bash que ve instal·lat amb el **SageMath**.
- Un **navegador d'arxius** gràfic. Tots els sistemes operatius moderns l'incorporen. En un Linux amb KDE, s'anomena *Dolphin*, en un MacOS, s'anomena *Finder*, en un Windows s'anomena *Explorador d'arxius*.
- Un **editor de text** bàsic. En un Linux amb KDE, podem fer servir *Kate*, en un MacOS, el *TextEdit*, en un Windows el *NotePad*.

### 1.1 Introducció al sistema operatiu GNU-Linux

Els ordinadors de la facultat tenen una versió de Linux instal·lada, que ens proporciona la millor manera d'accedir a **Bash** en aquests ordinadors. En les primeres seccions de les pràctiques parlem de l'escriptori KDE per a que us familiaritzeu amb el Linux de la facultat. No obstant, podem també executar **Bash** en qualsevol altre sistema, i només canviarà lleugerament el comportament de l'escriptori que descrivim a l'inici, cosa que no afecta al contingut principal del curs.

El terme **GNU-Linux** (o simplement **Linux**) no es limita a un sistema operatiu d'escriptori, sinó que es pot trobar a molts entorns. Vegem algunes definicions:

- El *Nucli del Linux* (*Linux Kernel*) el nucli s'un sistema operatiu *tipus Unix* lliure i de codi obert. La primera versió va ser programada per en Linus Torvalds al 1991. Actualment és un projecte amb la llicència *GNU* desenvolupat per programadors de tot el món. Nuclis derivats del nucli de Linux s'utilitzen a molts dispositius: ordinadors, telèfons mòbils, tauletes, discs durs multimèdia, maquinària de control numèric, ...

- Una *distribució de Linux* és un sistema operatiu que utilitza el *nucli de Linux* i hi afegeix el programari necessari per al funcionament d'un ordinador. Entre el programari inclòs hi acostuma a haver un *escriptori*, *navegador d'arxius*, *editor de text*, *processador de text*, *navegador d'internet*, ... Aquesta distribució pot ser lliure (per exemple *Debian* o *Ubuntu*) o comercial (per exemple, *Red Hat*). També hi ha distribucions que depenen d'altres distribucions (per exemple, *LinuxMint* depèn d'*Ubuntu*, alhora que *Ubuntu* depèn de *Debian*).
- El *Bash* és un *intèrpret d'ordres* utilitzat, entre altres, pel sistema *GNU-Linux*. Un intèrpret d'ordres, també anomenat *emulador de terminal* o *consola*, té la capacitat de comunicar ordres dels usuaris (introduïdes per teclat) al sistema operatiu i al conjunt de programes instal·lats en un ordinador, i presentar en forma de text en pantalla les respostes del sistema operatiu i altres programes. El *Bash* és un intèrpret d'ordres que segueix l'estàndard conegut com *POSIX*,<sup>1</sup> i en general el contingut d'aquest curs, on veurem algunes de les instruccions d'aquest intèrpret, serà aplicable en qualsevol sistema que s'adapti a aquest estàndard.


La forma com executem el Linux dependrà de l'entorn on treballem: podrà ser d'una forma completament gràfica amb un escriptori que permet interactuar mitjançant el ratolí, o podrà ser mitjançant l'accés a un terminal de text, on l'ordinador només entendre les instruccions que hi entrem mitjançant el teclat.

Aquest curs és de *Bash* i per tant s'hauria de poder entendre independentment de com l'executem (de forma gràfica o tant sols amb terminal text).

### 1.1.1 Iniciant una sessió en GNU-Linux als ordinadors de la facultat

Reinicieu l'ordinador i, a la pantalla de tria del sistema operatiu, cliqueu *Iniciar Linux*. Per a entrar heu de posar el vostre N.I.U. i la vostra paraula clau.

**Nomenclatura:** D'ara endavant

- *<NIU>* denota el vostre NIU.
-  denota la icona que hi ha a la cantonada inferior esquerra de la pantalla (logo del sistema).
- El directori *Home* és: */home/samba/homes/<NIU>* (normalment, en un ordinador personal, seria */home/<NIU>*). Els fitxers d'aquest directori es guarden de manera local a l'ordinador on esteu i (possiblement) s'esborren cada cop que entreu, pel que no és aconsellable deixar arxius que vulgueu conservar en aquest espai.
- El *directori personal* és un espai permanent d'emmagatzematge al que podeu accedir des de tots els ordinadors, tant amb GNU-Linux com amb Windows, i des de casa. Des del GNU-Linux de l'aula s'hi accedeix des de l'escriptori seguint *SERVER/HOME*. Si esteu en una terminal (consola) la instrucció és *cd /home/samba/homes/<NIU>/SERVER/HOME*.

<sup>1</sup>Els sistemes operatius *macOS* d'Apple també usen Bash, i per tant us permetran fàcilment seguir aquest curs, però el sistema *Windows* usa un intèrpret d'ordres no compatible amb *POSIX*.

### 1.1.2 Iniciant una sessió en GNU-Linux remota per Cocalc

El [Cocalc](#) és una web que ofereix software per a ser utilitzat de forma remota. Entre el software que hi ofereix hi ha el [LaTeX](#), [SageMath](#), [Python](#), [Terminal Linux](#) o un [Escriptori X11](#). En aquest cas, ens interessa iniciar un dels dos últims, i en aquest paràgraf comentem l'[Escriptori X11](#).

Si obrim un projecte nou que tingui un d'aquests terminals, ens obrirà una finestra amb un apartat que té un llistat de programari disponible, dels que destaquem:

- El [Gedit](#) és un editor de text. És el que recomanem per a fer les pràctiques d'aquesta assignatura des de Cocalc.
- L'[RStudio](#) i el [TeXStudio](#) són els programes que s'utilitzen en aquesta assignatura per a la part d'[R](#) i de [LaTeX](#).

També veiem un apartat que té un [Terminal](#). Aquí és on podrem escriure les instruccions de [Bash](#). En aquest cas, l'usuari és diu [user](#) i el seu directori *Home* és el [/home/user](#). També podem veure que aquest directori només hi ha els documents de treball que ja haguem creat, per tant, possiblement està buit i no té cap de les carpetes habituals (ni la [Documents](#), no [Escriptori](#),...).

Finalment, podem navegar entre les carpetes que tenim mitjançant un menú [Files](#).

### 1.1.3 Des d'una instal·lació pròpia o una màquina virtual

Si disposeu d'una instal·lació [Linux](#) en el vostre ordinador, us trobareu una situació semblant a la de l'apartat 1.1.1, però potser amb un escriptori o entorn gràfic amb un aspecte diferent. Comproveu en aquest cas quin és el programa corresponent a un terminal (cerqueu [terminal](#) a les aplicacions i, per exemple, si esteu en entorn KDE trobareu [Konsole](#)) i quin editor de text té instal·lat (cerqueu [editor](#) a les aplicacions i, per exemple, si esteu a l'entorn KDE trobareu [Kate](#)). Podeu seguir la pràctica com si estiguéssiu als ordinadors de la facultat, aplicant aquestes diferències (és a dir, cada cop que trobeu [/home/samba/homes/<NIU>](#) en el text, ho heu de canviar per [/home/nomusuari](#), etcètera.)

Si trebal·leu en un ordinador d'Apple, en el sistema operatiu [macOS](#) disposeu també de [Bash](#) de forma automàtica, ja que és el terminal per defecte d'aquest sistema. La situació és semblant a la de l'apartat 1.1.1, i les diferències bàsiques són:

1. Per obrir el [Bash](#), cal fer clic a la icona del Launchpad del Dock, escriure Terminal al camp de cerca i, després, fer clic a Terminal.
2. L'editor de text recomanat és el [TextEdit](#).
3. El directori *Home* de l'usuari és [/Users/Nomusuari](#) (on NomUsuari és el teu nom d'usuari.)
4. El navegador d'arxius és el [Finder](#).

Podeu seguir la pràctica com si estiguéssiu als ordinadors de la facultat, aplicant aquestes diferències (és a dir, cada cop que trobeu [/home/samba/homes/<NIU>](#) en el text, ho heu de canviar per [/Users/Nomusuari](#), etcètera.)

### 1.1.4 Bash en un dispositiu mòbil

També és possible utilitzar el *Bash* en tauletes o mòbils amb sistema operatiu *Android* o *iOS*, ja que aquests sistemes operatius compleixen essencialment els requeriments de l'estàndard POSIX. En el cas d'Android, us podeu baixar l'App *termux* de la Google Play Store. En els dispositius d'Apple, una possible App s'anomena *OpenTerm*.

Les instruccions que treballarem en aquest curs es poden practicar gairebé totes,<sup>2</sup> en cas de necessitat, en aquestes versions de *Bash* per a dispositius mòbils. En aquest cas, hi haurà canvis anàlegs als explicats per al cas de *macOS*, però més importants: el directori *Home* serà diferent (el podeu esbrinar escrivint l'ordre *pwd*) i, a causa del sistema de permisos, molt bona part de l'arbre de directoris no serà accessible (si es vol accedir a fitxers de les aplicacions del dispositiu, com fotografies, o veure al navegador d'arxius el directori *Home* i el que hi tenim, cal fer modificacions a la configuració). Recomanem fer les pràctiques sempre que es pugui en un ordinador.

## 1.2 Primers passos

Observeu que a la pantalla hi ha una barra inferior, amb moltes icones. Aquesta barra es diu el *plafó*, i conté molts elements. A l'esquerra de tot es troba el menú **K**, que conté tots els programes (o aplicacions), repartits en diferents categories. Dediqueu uns moments a localitzar els programes *firefox*, *kate* i *konsole*.

A la dreta del menú **K** es troba una zona que permet fer servir diversos escriptoris diferents a la mateixa sessió per a tenir les finestres més ordenades. A la dreta hi ha un rellotge, i d'altres icones. Aquesta disposició no està fixada *de fàbrica*, sinó que podem variar el contingut i la posició dels objectes (o estris). Investigueu, clicant-hi amb el botó dret del ratolí. Noteu, però, que els canvis que fem a la configuració del plafó i de l'escriptori es queden en un fitxer de configuració local de l'ordinador, així que si canvieu d'ordinador (o el reinstal·len) no trobareu la mateixa configuració.

### Navegador d'arxius

1. Obriu el navegador d'arxius (*dolphin*, *Finder*, *Explorador d'arxius...*) al vostre directori *home* (als ordinadors de la facultat, */home/samba/homes/<NIU>*), clicant la icona adequada i entrant a les carpetes corresponents. Als **Linux** de la facultat, si polseu la tecla **F9** fareu que aparegui/desaparegui a l'esquerra l'arbre dels directoris.
2. Creeu un directori que es digui *prac1* (cal tenir present que al sistema d'arxius del GNU-Linux les lletres majúscules són diferents de les minúscules), clicant amb el botó dret del ratolí i seleccionant l'opció **Crea nou → Carpeta**. Entreu-hi.
3. Creeu un fitxer nou que es digui *<NIU>.txt*, clicant amb el botó dret del ratolí i seleccionant el menú **Crea nou → Fitxer de text**. Esborreu l'arxiu que acabeu de crear, arrossegant-lo a la paperera.
4. Torneu al directori anterior (fent clic a la fletxa a dalt).

<sup>2</sup>Les instruccions *man* i *info* no estan disponibles d'entrada, tot i que al *termux* s'hi poden instal·lar addicionalment; l'*OpenTerm*, com l'interpret d'ordres de *macOS*, no té el *tac*.

### 1.2.1 Navegació amb terminal

Per a canviar de directori s'utilitza la instrucció `cd` i per a veure el contingut d'un directori la instrucció `ls`.

1. Obriu una consola (en Linux amb KDE, el programa `konsole`; en MacOS, el `Terminal`; en Windows, obriu el `jupyter` de SageMath, i obriu `New → Terminal`). Executeu `pwd` ("print working directory") per veure a quin directori esteu.
  - **Observació.** Si feu servir el Terminal de SageMath, veureu que el vostre directori `home` s'anomena `/home/sage`. Aquest directori s'identifica virtualment com el directori del sistema Windows que heu especificat com a directori inicial de SageMath, que pot ser `C:\Users\NomUsuari`.
2. Aneu al directori `home`, mitjançant la instrucció `cd /home/samba/homes/<NIU>`. Creeu un altre directori, que es digui `altredir`, amb la instrucció `mkdir altredir` i un fitxer nou, amb `touch altrefitxer.txt`. Feu `ls -l` per veure que efectivament heu creat el fitxer i el directori (observeu que la línia corresponent a `altredir` comença amb una `d`).
3. Comproveu a la finestra de `navegador d'arxius`, que hem fet servir anteriorment, que efectivament s'han creat el directori i el fitxer (si cal s'actualitza prement `F5`).
4. Esborreu tant el fitxer com la carpeta, mitjançant les instruccions `rm altrefitxer.txt` i `rmdir altredir`. Torneu a crear el fitxer `<NIU>.txt`.

**Tanqueu la consola que teniu oberta i obriu-ne una altra.** Per "navegar" pels directoris usarem principalment la consola. Per tant, ens hem d'acostumar a les seves instruccions. Executeu, successivament<sup>3</sup>

```
cd .
pwd
cd Escritorio
pwd
cd
pwd
cd Escritorio/..
pwd
cd /home/samba/homes/<NIU>/prac1 ; pwd
cd /home/samba/homes/<NIU>/prac1/../../../../ ; pwd
cd /usr; pwd
cd ~ ; pwd
```

<sup>3</sup>Aquestes instruccions suposen que:

- hi ha una carpeta anomenada `Escritorio` al vostre `home`. En cas de no ser-hi, podeu canviar les línies on surt `Escritorio` per alguna altra carpeta que tingueu al vostre `home`.
- Si no sou a una aula de la facultat heu de substituir `/home/samba/homes` per `/home` i `<NIU>` pel nom de l'usuari (`user` al Cocalc).

```
cd /etc/X11; pwd
cd $HOME ; pwd
cd -; pwd
```

### Notes:

1. La primera instrucció acaba en “.”.
2. El caràcter `~` equival sempre a *Home* (aquí `/home/samba/homes/<NIU>`) i s'obté prement simultàniament les tecles `Alt Gr` i `Ñ`).
3. El “fitxer” `.` (punt) denota el directori actual, sigui quin sigui, i el “fitxer” `..` denota el directori del nivell immediatament superior.
4. `$HOME` és una variable d'entorn que té valor `/home/samba/homes/<NIU>`, però li podem canviar el valor.
5. Podeu obrir una consola al directori que esteu veient amb el `navegador d'arxius` mitjançant la tecla `F4`.

### Exercici 1.1

Què passa quan es posen dues (o més) instruccions a la mateixa línia separades per “;” com al final del bloc d'instruccions anterior?

### Exercici 1.2

Perquè la primera instrucció (`cd .`) i la setena (`cd Escritorio/..`) no fan res?

### Exercici 1.3

Trobeu les tres instruccions `cd` del bloc anterior que són equivalents, *independentment* del directori on estigueu situats. Què fan?

### Exercici 1.4

Aneu a la vostra *Home* (executant `cd` sense cap altre argument). A la següent llista de instruccions, n'hi ha diverses que fan el mateix. Agrupeu les instruccions equivalents<sup>4</sup>.

---

<sup>4</sup>Aquestes instruccions suposen que:



```
ls
ls Escritorio/..
ls .
ls ..
ls /home/samba/homes/<NIU>/SERVER/HOME
ls ~/Escritorio
ls $HOME/Escritorio
ls /home/samba/homes/<NIU>/SERVER/HOME/prac1
ls /
ls /home/samba/homes
```

### 1.3 Obtenció d'ajuda

Gairebé totes les instruccions del GNU-Linux tenen una o més formes d'obtenir ajuda sobre el seu ús. La més ràpida, generalment, és **instrucció** `--help`, que dona un breu resum de les opcions disponibles. La forma més àmplia és fent servir les pàgines de **info**, que inclouen manuals breus i d'altres més extensos.

Si executeu `ls --help` | `less` (el signe “|” s’obté amb **AltGr** **1**), veureu que hi ha una llarga llista d’opcions que serveixen per fer que la instrucció `ls` ens obeeixi. Aneu amunt i avall amb les tecles de direcció, i sortiu prement **q**.

Si executeu `info`, en canvi, entrareu en un programa força complex per veure manuals, que es diu justament `info`. També us podeu moure a dintre del manual amb les fletxes i sortir amb **q**. A moltes distribucions modernes `ls` té un manual més complert, al que es pot accedir amb la instrucció `info coreutils ls`.

La pàgina inicial de l’`info` conté la llista de tots els manuals instal·lats, i es pot accedir a qualsevol d’ells desplaçant-se (amb les fletxes) a sobre del nom del manual i prement l’**Intro**. Cada manual pot contenir diferents capítols, que estan marcats amb el signe “\*”, i s’hi pot accedir també movent-s’hi a sobre amb el cursor i prement l’**Intro**. Per tornar enrere, només cal premer la tecla **l** (de *last*, “anterior”).

#### Exercici 1.5

Executeu `man gzip`. Quina diferència hi ha entre el que esteu veient i el contingut del manual que s’obté amb `info gzip`?

- hi ha una carpeta anomenada `Escritorio` al vostre *home*. En cas de no ser-hi, podeu canviar les línies on surt `Escritorio` per alguna altra carpeta que tingueu al vostre *home*.
- Si no sou a una aula de la facultat heu de substituir `/home/samba/homes` per `/home` i `<NIU>` pel nom de l’usuari.

### Exercici 1.6

A partir de la informació del manual de `ls` i de l'efecte de les mateixes instruccions, deduiu què fan la instrucció `ls *txt`, la instrucció `ls -ohX` i la instrucció `ls -mQu`.

### Exercici 1.7

Trobeu la forma, fent servir el manual de `info` (`man info`), de bolcar a un fitxer el contingut d'un capítol d'un manual. Bolqueu al fitxer `core.txt` un capítol del manual de `coreutils`.

## 1.4 Edició de fitxers

Un programa que permet editar (no processar) fitxers és el `kate` i un altre és el `gedit`. En els entorns KDE, podeu obrir el `kate` buscant-lo al menú **K** o bé des de la consola. A més, des de la consola, la instrucció `kate nomdelfitxer &` permet obrir el fitxer desitjat. També hi ha editors que viuen a dintre de la consola, per exemple l'editor `nano`, que és molt senzill però és útil si esteu treballant amb algun servidor remot i només disposeu de la consola. Obriu el fitxer `<NIU>.txt` que havíeu creat anteriorment, i escriviu-hi la següent informació:

```
Nom: El vostre nom
NIU: El vostre NIU
Grup: El vostre grup de pràctiques
Data: La data actual
```

Guardeu el fitxer.

Si només volem llegir un fitxer, no cal obrir-lo amb un editor, podem més senzillament fer que l'ordinador ens mostri el seu contingut. Aneu a una consola, navegueu fins el directori on es troba el fitxer que acabeu d'editar i executeu `cat <NIU>.txt` i `less <NIU>.txt`. Feu el mateix amb un fitxer més llarg, com ara el fitxer `/etc/X11/rgb.txt` (un fitxer de configuració de l'ordinador).

### Exercici 1.8

Quina diferència hi ha entre les instruccions `less` i `cat`?

**Observació:** Veurem aquestes instruccions amb més detall a la Secció 2.

## 1.5 Compresió de fitxers

Per comprimir un o diversos fitxers en un arxiu (d'ara endavant farem servir sempre la diferència de notació: *fitxer* = document o text; *arxiu* = resultat d'una compressió) es fa servir el programa *tar*. Si bé el *tar* és un programa extremadament complex (el seu manual té més de 150 capítols) el seu ús bàsic és força senzill.

Per comprimir uns fitxers (o directoris) a un arxiu, només cal executar la instrucció

*tar cvzf nomdelarxiu.tar.gz* llista dels fitxers i directoris per comprimir  
on les opcions *cvzf* indiquen:

- c** Creació de l'arxiu. En alternativa es poden afegir fitxers a un arxiu existent fent servir la opció *a* en lloc de *c*;
- v** Verbositat. Indica que el programa ens ha de llistar tots els fitxers que està compriment.
- z** Compresió (de tipus *gzip*, reflexat també pel fet que l'arxiu acaba per *.gz*. Si s'omiteix, l'arxiu es crearà sense compressió. També es poden fer servir altres tipus de compressions, com ara la de *bzip2*, amb opció *j* i arxiu acabat en *.bz2*, que comprimeix molt més, però és menys coneguda.
- f** Indica que el que ve just després és el nom que li volem donar a l'arxiu. No es pot omitir en l'ús bàsic.

En canvi, per descomprimir un arxiu existent es fa servir la instrucció següent:

*tar xvzf nomdelarxiu.tar.gz*

i els fitxers i directoris es crearan dintre del directori actual. Si no esteu segurs del contingut d'un arxiu, sempre és aconsellable crear un directori nou i descomprimir l'arxiu a dintre, per evitar confondre els nous fitxers amb els antics. En realitat no és necessari especificar el tipus de compressió a utilitzar en fase de extracció, ja que es pot deduir des de l'extensió.

Per conèixer el funcionament precís del programa *tar* és imprescindible fer servir el manual. A altres distribucions de GNU-Linux, el manual de *tar* conté més de 150 capítols, però el que està instal·lat a la sala d'ordinadors solament en té una desena. Això significa que només cobreix l'ús bàsic del programa.

Per veure el manual complet del programa *tar* baixeu-vos el fitxer auxiliar de la pràctica (que es diu *tar.info.gz*) des del campus virtual i guardeu-lo al directori que més us convingui. Seguidament e executeu en una consola la instrucció:

*info -f tar.info.gz*

(haureu de ser al directori on heu baixat el fitxer).

**Observació:** També podeu comprimir fitxers i directoris amb el programa gràfic *ark* (si està instal·lat) i, depenent també de la instal·lació, amb els menús que ofereix el botó dret del ratolí des del *dolphin*.

## 2 Manipulació de fitxers

### 2.1 Sistema de directoris. Crear, moure i esborrar

Hem vist a la Secció 1 que hi ha un sistema de directoris o carpetes, on cada directori se separa del superior i inferior mitjançant el caràcter */* (barra de dividir). També hem vist com moure'ns pels directoris amb la

instrucció `cd`.

Per a crear directoris i fitxers en blanc mitjançant la consola s'utilitzen les instruccions `mkdir` i `touch` respectivament (secció 1).

## Exercici 2.1

Creeu al vostre directori de treball *home* un directori que es digui `pract2`, i entreu a aquest directori. Executeu la instrucció `pwd` per a comprovar que esteu a aquest directori.

Per a copiar fitxers i directoris s'utilitza la instrucció `cp`. Podeu consultar el seu funcionament amb el manual de la instrucció.

## Exemple 2.1

Si volem copiar el directori `/etc/apt` al directori actual hem d'executar `cp -R /etc/apt .`

### Observacions

1. El punt del final de la instrucció és necessari. La instrucció `cp` interpreta l'últim directori (o fitxer) que entrem com el directori (o fitxer) destí.
2. L'opció `-R` és necessària sempre que vulgueu copiar directoris, i vol dir *recursiu*.
3. Podeu veure que ha creat un directori `apt` al directori on som.
4. Possiblement han sortit missatges d'error quan intentàvem copiar fitxers dels que no tenim permís de lectura.

## Exercici 2.2

Des del directori `pract2`, executeu la instrucció `cp -R /etc/apt/* .`

Què s'ha copiat ara? Què vol dir l'asterisc de la instrucció?

Per a moure fitxers tenim la instrucció `mv`. El seu funcionament és molt semblant a la instrucció `cp`.

## Exemple 2.2

Si anem al directori de treball *home* i volem moure el directori `pract2` a `prac2` tant sols hem d'escriure: `mv pract2 prac2`

### Exercici 2.3

Si encara no heu esborrat el directori `apt` que ara ha d'estar al directori `prac2`, expliqueu què fa la instrucció `mv prac2/apt`.

La mateixa instrucció serveix per a canviar el nom d'un fitxer.

### Exemple 2.3

Si volem renombrar el fitxer `sources.list` a que hi ha al directori `apt` al nom `llista` podem fer:

```
cd apt
mv sources.list llista
```

Si el que volem és esborrar fitxers tenim la instrucció `rm`, on a la utilització hem de dir el llistat de fitxers i directoris que volem esborrar. Per a esborrar directoris també hi ha la instrucció `rmdir`, però aquesta exigeix que el directori que volem esborrar estigui buit.

**Observació important:** La instrucció `rm` esborra de manera permanent el que li diguem. No ho envia a cap part de l'ordinador en que es pugui recuperar. A més, segons com estigui configurat no demana confirmació abans d'esborrar.

### Exercici 2.4

Des del directori `home` esborreu el directori `apt` i tot el seu contingut. Proveu-ho primer amb la instrucció `rmdir` (que no hauria de funcionar ja que el directori no està buit) i finalment feu-ho amb la instrucció `rm` (podeu consultar l'ajuda per a veure com es pot esborrar de manera recursiva d'un directori cap endavant).

## 2.2 Entrada i sortida. Les “pipes”

Els programes de consola GNU-Linux (gairebé) *sempre* tenen una entrada estàndard (`standard input-stdin`), que està connectada al teclat, i una sortida estàndard (`standard output-stdout`), que està connectada a la pantalla.

Hi ha una forma comú de connectar la sortida d'un programa a l'entrada d'un altre, fent servir el connector “pipe”: `|`. D'aquesta manera es formen unes “pipes” de dades, que passen per l'acció de diferents “motors” (programes).

Treballarem diversos exemples perquè pugueu veure la importància de les “pipes”. Necessitem un fitxer llarg que podem crear amb la instrucció `ls -l /usr/bin > llarg.txt`. Podem tenir el fitxer obert amb la instrucció `less` per a veure el seu contingut.

1. Entreu al directori `prac2` que heu creat abans.

2. Executeu `cat llarg.txt`.
3. Executeu `cat llarg.txt | tac`. Aquesta instrucció imprimeix l'entrada en ordre invers, des de l'última línia a la primera.
4. Executeu `cat llarg.txt | nl`; aquesta instrucció renumera les línies.
5. Executeu `cat llarg.txt | nl | tac` i després `cat llarg.txt | tac | nl`. Com veieu, invertint l'ordre dels factors (els motors) canvia el resultat!

**Nota:** A la terminal de l'**OSX** no hi ha implementada la instrucció `tac`. Podeu usar `tail -r` en el seu lloc.

Torneu a executar les instruccions anteriors afegint `| less` al final, per poder desplaçar-vos més fàcilment pel text de sortida de la instrucció.

No només existeixen aquestes instruccions, n'hi ha moltes més. A un GNU-Linux basat en Debian (per exemple, l'Ubuntu) podeu trobar una llista de instruccions bàsiques executant `info coreutils`. Tots aquests programes es poden encadenar i les dades passaran d'un a l'altre com en una cadena de muntatge.

Un altre programa típic que es fa servir amb “pipes”, generalment al final, és el `wc` (de *word count*). Executar `man awk | wc` donarà un recompte del nombre de línies, paraules i caràcters que componen el manual de l'`awk`.

No us explicarem totes les intruccions possibles, sinó que intentarem donar les eines necessàries per a poder connectar programes. Només explicarem en detall el funcionament de tres programes: el `grep`, el `sort` i l'`awk`.

Una altra forma de “pipe” és la que deriva la sortida d'un programa cap a un fitxer, fent servir el connector “>” (o “>>” si volem afegir les dades al final del fitxer). També es pot derivar el contingut d'un fitxer cap a l'entrada d'un programa (amb `programa < fitxer`), encara que és totalment equivalent a `cat fitxer | programa`.

Una altra eina important quan estem fent servir “pipes”, és la “T”, que és un “canal secundari” pel qual podem tenir una còpia de les dades que estan passant per la “pipe”. Com sempre al GNU-Linux, el nom d'aquesta eina és senzill, no és més que el nom anglès de la T, `tee`. Fent

```
...intruccions | tee UnFitxer | més intruccions
```

obtindrem una còpia de les dades, passant per la “pipe”, al fitxer que es diu *UnFitxer* sense afectar de cap forma al funcionament de les altres intruccions. Per exemple

```
man awk | tee awk.txt | wc
```

dóna, com abans, un recompte del nombre de línies, paraules i caràcters que componen el manual de l'`awk`, però a més tindrem una còpia del manual de l'`awk` al fitxer `awk.txt`.

## 2.3 Ordenació de dades: `sort`.

A vegades necessitem ordenar unes dades per algun ordre (alfabètic, numèric, etc.). Si aquestes dades es troben a un fitxer de text podem fer servir la instrucció de consola `sort`, que vol dir “ordena”.

Per ordenar un fitxer només cal afegir al final de tot el nom del mateix fitxer: `sort fitxer`. S'obté el mateix resultat fent-lo servir com a “pipe” de la instrucció `cat`: “`cat fitxer | sort`”. A més a més, si

volem que les dades ordenades vagin cap a un fitxer, ho hem de fer explícitament, amb un “pipejat” cap a un fitxer (>).

L'ús habitual de `sort` és el següent:

```
...| sort opcions clau1 clau2 ...| ...
```

on cada *clau* indica la columna o columnes que volem fer servir per a ordenar, i les opcions indiquen quin tipus d'ordenació volem. Per exemple, per a ordenar un fitxer alfabèticament, però en el qual hi ha una primera columna (per exemple el NIU d'una persona) seguida del nom, haurem de fer

```
cat fitxer | sort -k2
```

Això vol dir que volem una ordenació alfabètica però fent servir de la columna 2 endavant. Per una ordenació numèrica, o bé afegim la opció “`-n`” o bé afegim la lletra *n* després de la clau. Per exemple, per ordenar per la columna 2 numèricament i, quan el valor de la columna coincideix, ordenar per la columna 4, alfabèticament, es pot fer servir

```
...| sort -k2,2n -k4,4
```

Les lletres per indicar la forma d'ordenar són les següents:

- n* ordena de forma numèrica. Per exemple, alfabèticament “10” va abans que “9” però numèricament va després.
- r* ordena a l'invers, per exemple de gran a petit, de desembre a gener o de “Z” a “A”, segons les altres lletres posades.
- b* indica que no s'han de considerar espais en blanc. D'aquesta forma les columnes només contenen les dades, sense els espais que les delimiten.
- d* indica que s'ha d'ordenar com si fos un diccionari, tenint en compte només lletres, nombres i espais. Altrament, tot caràcter es pren en consideració.
- f* indica que no s'han de considerar majúscules i minúscules com diferents. Aquest tipus d'ordenació té problemes en algunes instal·lacions de GNU-Linux, ja que de vegades no es reconeix que “ç” és la versió minúscula de “Ç”.

Tota lletra pot servir tant com a “opció” per una clau, o com a “opció general”, si li afegim un guió abans.

## Exercici 2.5

Quin tipus d'ordenació representa la instrucció

```
sort -k4,4nr -k2,2f?
```

Doneu una estructura de fitxer de dades on es pugui aplicar.

## Exercici 2.6

Doneu el llistat de fitxers del directori `/usr/bin` ordenats de mes vell a més nou en una sola instrucció.

## 2.4 Sistema de fitxers: permisos

El *GNU-Linux* és un sistema multiusuari i té un sistema de fitxers que permet garantir el funcionament del sistema i la privadesa de tots els seus usuaris. Molts dels *servidors* utilitzen aquest sistema operatiu i tenen molts usuaris que treballen a l'hora en aquell ordinador. A més hi ha un usuari (en general s'anomena *root*) que té poder per administrar la màquina i per tant per modificar, llegir o esborrar totes les dades del sistema.

Donant els permisos convenients als fitxers i directoris evitem que un altre usuari del sistema pugui veure o esborrar els nostres fitxers<sup>5</sup>.

Per a veure els permisos d'un fitxer o directori utilitzem la opció `-l` de la instrucció `ls`, i mirem la primera columna de les dades.

### Exemple 2.4

Mirem la sortida de:

```
ls -l /usr/bin
```

veiem els permisos de cada fitxer/directori a la primera columna.

El primer caràcter (un guió per la majoria) ens diu si es tracta d'un fitxer (guió), un directori (una lletra *d*) o un enllaç (una lletra *l*).

Llavors venen nou caràcters que s'agrupen de 3 en 3. Cada cadena de 3 està format per *rw*~~*x*~~ (*r*ead, *w*rite, *x*ecute), o bé substituint algunes d'aquestes lletres per guions. El primer bloc de 3 lletres afecta al propietari del fitxer (en aquest cas *root*). El segon bloc al grup (en aquest cas torna a ser *root*, però podria ser diferent) i el tercer bloc a la resta d'usuaris.

### Exemple 2.5

La cadena *rw*~~*x*~~*r-x---* correspondria a un fitxer on el propietari pot llegir, escriure i executar el fitxer; els usuaris del grup assignat poden llegir-lo i executar-lo i la resta d'usuaris no poden ni llegir-lo, ni escriure-hi, ni executar-lo.

Per a canviar els permisos d'un fitxer o directori tenim la instrucció *chmod*. Una de les sintaxis per executar aquesta comanda és assignant un número del 0 al 7 a cada bloc, i assigna els permisos depenent de com és en base 2. A la taula següent hi ha el número en base 10, en base 2 i els corresponents permisos:

<sup>5</sup>Això depèn de com s'ha formatat el dispositiu. Per exemple una clau USB s'acostuma a formatar en sistema *FAT32* que no admet permisos: qualsevol usuari pot esborrar els fitxers.



Base 10	Base 2	Permisos
0	000	---
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx

Ara l'ús de la instrucció `chmod` es redueix a dir quins permisos volem per *usuari-grup-tots* en tres dígit seguits entre el 0 i el 7, seguit del fitxer o directori del que volem canviar els permisos.

### Exemple 2.6

Si tenim el fitxer `hola.txt` i volem que només nosaltres el puguem veure i modificar executem:

```
chmod 700 hola.txt
```

### Exercici 2.7

Quina opció permet canviar els permisos de tots els fitxers i directoris d'un en endavant?

### Exercici 2.8

La lletra `x` del les inicials `rwx` en el cas dels directoris no vol dir executar (els directoris no s'executen). Esbrineu què vol dir.

Creeu un directori nou i copieu un fitxer qualsevol dins. Doneu els permisos `rw-rw-rw-` al directori. Podeu entrar al directori? I veure el fitxer que conté?

Ara doneu els permisos `-wx-wx-wx` al mateix directori. Podeu entrar? I llistar el contingut?

## 3 Edició i manipulació de fitxers

### 3.1 `grep` i expressions regulars

El programa `grep` serveix per a extreure les línies que compleixen una condició donada d'entre les línies d'un fitxer o, també, de la sortida estàndard d'una instrucció anterior.

En primer lloc estudiarem el programa `grep` per a processar la sortida estàndard d'una instrucció anterior (és a dir *com a filtre*). Per exemple, podem llistar tots els fitxers del directori `/usr/bin` que continguin la

cadena “*zip*” al seu nom ho podem fer amb la instrucció

```
ls /usr/bin | grep zip | less
```

Durant la primera pràctica varem executar la instrucció `ls *txt`. La cadena “*\*txt*” és un cas particular del que es diu *expressió regular*, que indica diferents paraules o frases. Per exemple, “*\*txt*” vol dir “qualsevol cosa que acabi amb *txt*” o, més precisament, “qualsevol cosa seguit de ‘*txt*’”.

La “paraula” buscada pel `grep` també és una expressió regular. Per tant, tot seguit farem un resum de les components d'una expressió regular.

•: Cada caràcter (per exemple *a*) es representa ell mateix.

*Rangs:* `[acds]` representa qualsevol caràcter en el rang escrit, en aquest cas representa o bé una *a*, o una *c* o una *d* o una *s*. Els rangs seguits es poden indicar amb un guió (`f-m` és el mateix que `fghijklm`). Per exemple `[0-9]` representa qualsevol dígit i `[A-La-l]` representa qualsevol caràcter entre *a* i *l*, tant majúscula com minúscula.

Executeu `ls /usr/bin | grep [w-z]ip` per veure el resultat.

Si afegim un accent circumflex al principi del rang (`[^a-z]`) representa que NO volem els caràcters del rang.

A dintre d'un rang, es poden donar conjunts preestablerts, per exemple `[:alpha:]` que és igual a `a-zA-Z`, però altres que no es podrien escriure d'altra manera, com ara `[:space:]`: els espais i tabulacions, o `[:print:]`: tots els caràcters que es poden veure a pantalla. Per exemple, `a[[:space:]][[:punct:]]a` representa dues lletres *a* separades per un espai o un signe de puntuació.

*Especials:* Per indicar l'inici de la línia es fa servir `^` (fora d'un rang!) mentre que `$` representa el final de la línia. `<` i `>` representen l'inici i el final d'una paraula. Executeu

```
ls -l /usr/bin | grep zip$
```

per trobar els fitxers que acaben en “zip”, i

```
ls -l /usr/bin | grep "<zip>"
```

per trobar els fitxers que es continguin “zip” com a paraula aïllada (el guió es considera que separa paraules).

*Repeticions:* `?` indica que l'objecte anterior (si cal, entre parèntesis) pot aparèixer o no, però no més d'una vegada. Per exemple `^(go)?ogle` correspon a les línies que comencen per *google* i per *ogle*.

`+` indica que l'objecte anterior ha d'aparèixer una o més vegades. Per exemple `(go)+ogle` correspon tant a *gogoogle* com a *google*, però no a *pirgle*.

`*` indica que l'objecte anterior pot aparèixer o no, i un nombre arbitrari de vegades. Per exemple `g(o)*gle` correspon a *ggle* com a *goooooooooogle*, però no a *gaagle*.

Podem expressar fins i tot la quantitat exacta de vegades que l'objecte ha d'aparèixer. `{5}` volem 5 repeticions. Si hi posem `{5,8}` vol dir que pot aparèixer de 5 a 8 vegades, mentre si posem `{7,}` vol dir que ha d'aparèixer al menys 7 vegades.

### Exercici 3.1

Què vol dir l'expressió regular següent?

```
^([[:alnum:]]+[[:space:]]*){3}[[:alnum:]](?:zip\>[[:print:]].*\<(des)?fer
```

Feu exemples de frases que corresponen a l'expressió i que no hi corresponen. En el segon cas, justifiqueu-ne la manca de correspondència.

L'explicació completa de les expressions regulars es troba al manual de *grep* (*man grep*).

**Important:** Per a que *grep* entengui les expressions regulars en tota la seva potència, és necessari fer servir la opció *-E*, i també posar les cometes al voltant de l'expressió regular.

Una altra opció interessant del *grep* és *-v*, que serveix per seleccionar les línies que NO es corresponen amb l'expressió donada. Per exemple *ls | grep -v [aA]* dona tots els fitxers sense la lletra "a" en el seu nom. Això també ho podeu fer sense fer servir expressions regulars amb la instrucció

```
ls | grep -vi a.
```

**Practicant amb *grep*:** Pels següents exercicis necessitareu el fitxer *observ.txt*. Creeu un directori anomenat *prac3* al vostre directori d'usuari permanent. Seguidament, baixeu-vos el fitxer auxiliar de la pràctica (*practica03-aux.tar.gz*) des del campus virtual, guardeu-lo i descomprimiu-lo al directori que acabeu de crear. Hi trobareu el fitxer *observ.txt* que conté una sèrie d'observacions d'uns investigadors.

1. Llisteu tots els arxius del directori */usr/bin*, seleccionant i mostrant només els que contenen la paraula *zip* al seu nom. La instrucció és *ls /usr/bin | grep --color=auto zip*.
2. El programa *cal* ens mostra el calendari d'un mes o un any arbitrari. Executeu *cal 12 2021* per veure el calendari del desembre.  
Seleccioneu la setmana que conté el vostre aniversari del 2030. Per fer-ho, heu d'executar *cal 7 2030 | grep "<25>"* si el vostre aniversari és el 25 de juliol.
3. Seleccioneu les línies del fitxer *observ.txt* que contenen la paraula "Lleó" al principi. Això es pot fer de tres formes:  
*cat observ.txt | grep "^Lleó"*  
*grep "^Lleó" < observ.txt*  
*grep "^Lleó" observ.txt*  
**Nota:** A la última forma *grep* no entra en una "pipe". Treballa directament sobre un fitxer. De fet és l'ús més habitual del *grep*.

4. De les línies anteriors, seleccioneu les que la segona columna (dia d'observació) és 2:  
*cat observ.txt | grep "^Lleó" | grep -E "[[:alpha:]]+[[:space:]]+2"*  
o bé  
*cat observ.txt | grep -E "^Lleó[[:space:]]+2"*

## Exercici 3.2

Construïu una expressió regular que, aplicada a la instrucció

```
ls -l | grep expressió
```

doni tots els fitxers que tenen data del mes actual, però de qualsevol any. Executeu aquesta instrucció aplicada a qualsevol directori de l'ordinador i bolqueu el seu resultat al fitxer anomenat *llista\_fitxers.txt*.

L'objectiu de l'*awk* és el d'executar una acció per cada línia de text d'un fitxer (o de l'entrada estàndard) que compleix una condició prefixada. Si no s'especifica cap acció s'imprimeix la línia llegida. Com a conseqüència d'això l'*awk* emula el *grep*:

```
awk '/expressió/' fa el mateix que grep expressió.
```

**Nota:** Segons quina instal·lació de GNU-Linux s'utilitzi, hi ha diverses versions de l'*awk* instal·lades. Usualment s'anomenen "awk", "mawk" o "gawk".

Suposem que volem escriure, de les línies del fitxer d'observacions, només les que es refereixen a lleons, però només volem saber les dates en que s'han fet les observacions (2na i 3ra columna). Per fer això hem d'escriure

```
awk '/Lleó/ {print $2, $3}' observ.txt
```

Aquest exemple ens mostra el patró general d'una instrucció per l'*awk*, que és:

```
'/expressió regular/instruccions'
```

i aquest patró general es pot repetir. Per exemple, la següent instrucció:

```
awk '/^Lleó/ {print "Lleons:", $3} /^Tigre/ {print "Tigres:", $4}' observ.txt
```

imprimirà per pantalla només les línies que comencin amb "Lleó" o "Tigre" i d'aquestes, respectivament, només la columna 3 o la 4. Com a cas especial, la "columna 0" (*\$0* indica tota la línia llegida).

En general, el funcionament de l'*awk* és el següent:

1. Agafa una línia de l'entrada (tan d'una "pipe" com d'un fitxer).
2. Revisa si aquesta entrada coincideix amb alguna de les expressions.
3. Per a cada expressió amb la qual l'entrada coincideix, executa les instruccions corresponents.

Hi ha dues expressions, especials que corresponen a l'inici i el final de l'entrada. Són, respectivament, *BEGIN* i *END*. Aquestes expressions no s'han d'incloure entre signes "/" ja que no són expressions regulars.

### 3.1.1 Expressions de l'*awk*

A part de la busca d'una expressió regular, l'*awk* pot comprovar condicions molt més complexes. Per exemple, pot veure si una certa expressió regular coincideix amb una certa columna de l'entrada. La instrucció

```
cat observ.txt | awk '$5 ~ /20/ {print}'
```

imprimeix les línies que contenen les lletres "20" a la cinquena columna. A més, es poden fer comparacions, com ara *\$1=="Tigre"* o *\$4<5*.

També es poden donar condicions sobre el nombre de línies que ja s'han mirat (una mena de comptador) amb la variable `NR` (Number of Rows) i sobre el nombre de columnes de la línia actual, amb la variable `NF` (Number of Fields). L'expressió `$i` fa referència a la columna `i` del fitxer mentre que `$0` fa referència a tota la línia (registre) llegida.

Executeu `awk '{print NR, $0}' observ.txt`. Com veieu, `NR` es pot utilitzar dins d'una instrucció. Executeu ara `cat observ.txt | awk 'NR==6 {print}'`, veureu que només s'imprimeix la línia 6 (compareu amb la sortida de la instrucció anterior).

### 3.1.2 Instruccions de l'`awk`

Per a cada expressió que coincideix amb la línia llegida, s'han de posar una o més instruccions que s'han d'executar. La més senzilla és `print`, que simplement imprimeix a la pantalla la línia. També es pot "pipejar" l'escriptura, fent servir els operadors `|`, `>` i `>>` com ja heu fet abans. Per exemple, la instrucció `awk '{print $1 >> "primeracolumna.txt"; print $2 >> "segona.txt"}' observ.txt` crea el fitxer `primeracolumna.txt` que conté la primera columna del `observ.txt` i el `segona.txt` que conté la segona.

Es pot instruir `awk` per que faci moltes més coses. Per exemple, la següent instrucció `awk 'BEGIN {c=0;p=0} {c=c+length($0);p=p+NF;} END {print c, p, NR}' observ.txt` escriu el nombre de caràcters, paraules i línies de l'entrada. És a dir, és equivalent a `wc observ.txt`.

En aquest exemple hem fet servir la suma. Podem utilitzar les quatre operacions bàsiques de l'aritmètica (`+`, `-`, `*`, `/`), i a més el "mòdul" `%`, que dóna la resta de la divisió entera, i la potència `^`. A més, hem introduït les variables "`c`" i "`p`", que anem canviant a cada lectura. A l'hora d'imprimir, una variable s'imprimeix com qualsevol altra cosa, posant-la després d'un `print`. També es poden imprimir cadenes de caràcters posant-les entre cometes.

Hi ha també una sèrie de funcions predefinides que es poden utilitzar. Per exemple:

**length:** Dóna la llargada del seu argument. Exemples: `var = length($0)` dóna a `var` el nombre de caràcters de la línia actual i `length($3)` dóna la llargada de la tercera paraula de la línia actual.

**split:** Separa l'argument en trossos amb el separador escollit. Exemple: `split($1,trossos,",")` guarda en `trossos[1]...trossos[5]` les diferents parts de `$1` separades pel caràcter `",`. Més precisament, si `$1` és `1,34,hola,t1,5666666`, llavors `trossos[1]=1`, `trossos[2]=34`, `trossos[3]="hola"`, `trossos[4]="t1"`, `trossos[5]=5666666`.  
`split($0,columna)` ja sabem que fa: aïlla els trossos de `$0`, que són `$1`, `$2`,..., però amb l'avantatge que després podem fer servir la notació `columna[1]` en lloc de `$1`.

Com veieu, un "programa" `awk` es pot allargar i complicar bastant. Tenim la possibilitat d'automatitzar l'`awk` fent que llegeixi les instruccions d'un fitxer, de forma que no cal escriure-les cada vegada. Això es fa amb la opció `-f`. Per exemple, feu servir la instrucció de l'`awk` que trobeu al material auxiliar: `awk -f filtre.awk observ.txt`

Obriu el fitxer d'instruccions `filtre.awk` amb l'editor.

### Exercici 3.3

Quin efecte creieu que té l'execució d'aquestes instruccions sobre el fitxer de dades? Comenteu el fitxer.

Com veieu: múltiples instruccions per al mateix bloc s'han de dividir mitjançant signes `;` i es poden posar comentaris, iniciant la línia amb un caràcter `#`.

### Exercici 3.4

Modifiqueu el fitxer `filtre.awk` per tal que imprimeixi la mitjana dels pesos dels exemplars observats per a cada grup. Guardeu el resultat com a `mitjanes.awk`. Podeu suposar que no hi ha més de 10 columnes.

**Suggeriment:** el nombre d'exemplars, per a cada línia, es troba a partir de la variable `NF`, restant-li el nombre de columnes que no contenen dades d'observació, el nom de la espècie i la data.)

Una altra aplicació de la instrucció `awk` és la de modificar arxius de dades en format text (per exemple els `csv`) que ens venen donats i que no s'adapten als nostres propòsits. Vegem-ne un parell d'exemples:

#### Exemple 3.1

El fitxer `dn02.csv` conté dues columnes separades per punt-i-coma i el decimal està entrat amb coma. La primera línia és un encapçalament en format text. Suposem que tant sols volem la segona columna sense l'encapçalament i amb els decimals separats per un punt. Podem executar:

```
awk 'BEGIN {FS=";"}; /^ [0-9]/{sub(/,/,".")1;print $2}' dn02.csv > segcol.csv
```

#### Exemple 3.2

Volem obrir el mateix fitxer `dn02.csv` des d'un programa que només entén les dades quan estan entrades entre claudàtors `[]`, separades per coma i amb el decimal un punt. Per exemple, la fila `1;-1,2932049266` s'ha de convertir en `[1,-1.2932049266]` i la resta com correspongui. Ho podem fer fent:

```
awk 'BEGIN {FS=";"}; {sub(/,/,".")1;print "["$1","$2"]}' dn02.csv > dn02_corchets.csv
```

Destacar d'aquests exemples que s'ha utilitzat la constant `FS` per a dir quin és el separador de camps i que el text que entrem entre cometes dins d'un `print` s'imprimeix de forma literal.

## Apèndix: La consola del Windows

El contingut d'aquest apèndix és merament informatiu. L'avaluació del bloc GNU-Linux de l'assignatura només tindrà en compte els apartats anteriors, en `Bash`.

Tots els sistemes operatius incorporen un programa *intèrpret d'ordres*, també anomenat *emulador de terminal* o *consola*, que, com el *Bash* que hem après, té la capacitat de comunicar ordres dels usuaris (introduïdes per teclat) al sistema operatiu i al conjunt de programes instal·lats en un ordinador, i presentar en forma de text en pantalla les respostes del sistema operatiu i altres programes.

La consola de Windows té una sintaxi una mica diferent del *Bash* i altres consoles de l'estàndard POSIX, tot i que els conceptes bàsics són els mateixos. En aquest apèndix expliquem breument la consola de Windows i per tant a partir d'ara suposarem que estem treballant amb un ordinador amb el sistema operatiu Windows de Microsoft.

Segons la versió de Windows que tenim instal·lada, podem tenir accés a consoles lleugerament diferents; en aquest document utilitzem la consola que està present en tots els sistemes Windows a data d'avui: el programa *cmd.exe*. És possible que en el teu Windows també hi hagi la *PowerShell*, que disposa d'algunes instruccions habituals en la consola de Linux que no existeixen en *cmd.exe* (per exemple, *ls/cp/mv*, *cat* o *tee*).

## Navegar per l'arbre de directoris

En Windows, cada unitat d'emmagatzematge, ja sigui un disc dur (més precisament, una partició) intern o extern, una clau USB, un CD, etc., té associada una lletra. Típicament *C:* correspon al disc dur principal (intern) on hi ha els arxius del sistema operatiu i els directoris dels usuaris. El caràcter *\* fa de separador de directoris. Així, *C:\Windows* és el subdirectori *Windows* de la unitat *C:*, i *C:\Windows\System.ini* és el fitxer *System.ini* del directori *C:\Windows*.

Obriu una consola<sup>6</sup> i executeu *cd* ("current directory") per veure a quin directori esteu. Típicament, el prompt del sistema també ens diu en quin directori estem, com ara

```
C:\Users\Nom Usuari>
```

## Exercici 3.5

Per a canviar de directori s'utilitza la instrucció *cd* i per a veure el contingut d'un directori la instrucció *dir*.

1. Aneu al directori *Documents*, mitjançant la instrucció *cd Documents* (ara, *cd*="change directory"). Creeu un altre directori, que es digui *altredir*, amb la instrucció *mkdir altredir*. Feu *dir* per veure que efectivament heu creat el directori (observeu que la línia corresponent a *altredir* conté l'expressió *<DIR>*).
2. Obriu el directori *Documents* en una finestra de l'*Explorador de fitxers* (és possible que en l'Explorador, aquesta carpeta es vegi amb el nom "*Mis Documentos*"), i comproveu que efectivament s'ha creat el directori *altredir*.
3. Esborreu la carpeta, mitjançant la instrucció *rmdir altredir*. Es pot esborrar una carpeta que té fitxers dins, junt amb el seu contingut, amb l'opció */s*: *rmdir /s altredir*.

<sup>6</sup>A la barra del Windows, on hi ha la lupa, escriviu *cmd* o *Símbol del sistema*.

Tanqueu la consola que teniu oberta i obriu-ne una altra. Executeu, successivament:

```
cd
cd .
cd Documents
cd ..
cd Documents\..
cd C:\Windows\Temp
cd C:\Windows\Temp\..\..
cd %userprofile%
cd C:\Users
cd C:\Users\%username%
```

Compareu la sintaxi amb el cas de Linux a la secció 1.2.1. En algunes versions de Windows podeu obrir una consola al directori que esteu veient en l'*Explorador d'Arxius* anant al menú *Fitxer*→*Obrir Consola*, on *Consola* pot ser el *Símbol del Sistema* o el *PowerShell* depenent de la configuració.

Hem vist que, com en *Bash*, per a crear i esborrar directoris mitjançant la consola de Windows s'utilitzen les instruccions *mkdir* i *rmdir* respectivament. Per a copiar fitxers i directoris s'utilitzen la instruccions *copy* i *Xcopy*. Podeu consultar el seu funcionament escrivint *help copy* o *help Xcopy* a la consola.

Dues altres instruccions bàsiques són *move* (moure o canviar el nom) i *del* (esborrar). Com abans, podeu usar la instrucció *help* per aprendre'n el funcionament. Igual que en *Bash*, les instruccions *rmdir* i *del* esborren de manera permanent el que li diguem.

Els programes *grep* i *awk* no formen part del sistema operatiu Windows de Microsoft. Sí que és possible obtenir-ne d'equivalents i instal·lar-los per poder realitzar les mateixes tasques que ara sabem fer en **Linux**.