Guia Shell Script

1. Conceptes bàsics

1.1. Variables

Variables locals: només són visibles per la shell on estem treballant, no són visibles per cap subshell, és a dir, no són visibles per cap subprocés de la shell.

Variables d'entorn: són visibles tant per la shell on estem com per qualsevol subshell que obrim (o per qualsevol subprocés de la shell)

j@j:~$ varmeva=3 j@j:~$ echo $varmeva 3 j@j:~$ bash j@j:~$ echo $varmeva j@j:~$ exit exit j@j:~$ export varmeva j@j:~$ echo $varmeva 3 j@j:~$ bash j@j:~$ echo $varmeva 3	declaro una variable local miro el seu valor entro en una subshell no té valor perquè és local tornem a la shell principal converteixo la var. local a var. d'entorn a la shell principal, puc veure el seu valor entro en una subshell ara sí que veiem el seu valor!!!

Variables predefinides: Disposem també d'una sèrie de variables ja definides que ens poden ser de gran ajuda per tal d'obtenir o desar informació genèrica.

j@j:~$echo "El meu directori de treball es $HOME i el meu id de grup es $GROUPS"
El meu directori de treball es /home/joel10olor i el meu id de grup es 1000

1.2. Substitució de variables

Substitució de variables: És la tècnica que farem servir per fer referència al valor contingut en una variable.

j@j:~$ saluda="hola" j@j:~$ nom="anna" j@j:~$ echo saluda nom saluda nom j@j:~$ echo $saluda $nom hola anna	defineixo variable saluda amb valor “hola” defineixo variable nom amb valor “anna” mostro per pantalla els valors de les variables... està malament: he posat els noms de les vars, no el seu valor ara està bé!

j@j:~$ masc="gat" j@j:~$ echo masculi:$masc femeni:$masca masculi:gat femeni: j@j:~$ echo masculi:$masc femeni:${masc}a masculi:gat femeni:gata	defineixo variable masc amb valor “gat” mostro el $masc i $masc seguit d'una a m'interpreta masca com una var (buida) poso {} diferenciant el nom de la variable això és el que volia!!!

1.3. Substitució de comandes

Substitució de comandes: És la tècnica que farem servir per substituir el nom d'una comanda per la sortida d'aquesta.

j@j:~$ whoami joel10olor j@j:~$ echo Soc el/la whoami Soc el/la whoami j@j:~$ echo Sóc el/la `whoami` Sóc el/la joel10olor j@j:~$ echo Sóc el/la $(whoami) Sóc el/la joel10olor	una comanda ens diu com a què estàs connectat/da si volem mostrar per pantalla i posem la comanda... ...ens surt literalment el que li posem substituïm whoami pel valor que retorna i... ...obtenim el que volíem!!! una altra manera de ferho

j@j:~$ date +%F; date +%D 2022-11-05 11/05/22 j@j:~$ aaaammdd="%F" j@j:~$ mmddaa="%D" j@j:~$ echo "Data (aaaa-mm-dd):`date +$aaaammdd`" Data (aaaa-mm-dd):"2022-11-05" j@j:~$ echo "Data (mm/dd/aa):`date +$mmddaa`" Data (mm/dd/aa):"11/05/22"	executem date amb formats %F i %D donem com a valors aquests formats a dues variables substituim comandes i vars…

j@j:~$ CS=/var/backup-`date +%Y%m%d`.tgz j@j:~$ tar -czf $CS /home/nomusu	variable amb el nom de la còpia de seguretat comprimim i empaquetem el directori de treball

1.4. Caràcters especials

Caràcters especials: Hi ha caràcters que per a la shell tenen un significat especial. Existeixen diferents tècniques per tal que la shell ignori aquest significat o el tingui en compte.

j@j:~$ echo "El "silenci"" El silenci j@j:~$ echo El \"silenci\" El "silenci"	volem mostrar silenci entre “”... ens interpreta les “ de silenci com a caràcter especial! les hem d'”escapar”

j@j:~$ echo 'Soc el/la $LOGNAME i estic a $PWD' Soc el/la $LOGNAME i estic a $PWD j@j:~$ echo "Soc el/la $LOGNAME i estic a $PWD" Soc el/la joel10olor i estic a /home/joel10olor j@j:~$ echo "Soc el/la $LOGNAME i \$PWD: $PWD" Soc el/la joel10olor i $PWD: /home/joel10olor	amb '' … ...no interpreta $ amb “” … ...sí que els interpreta! barrejant “” i \ … ...podem interpretar o no

1.5. Redireccionament e/s

stdin, stdout i stderr:
Hi ha comandes que accepten les dades d'entrada pel que anomenem entrada estàndard, que és el teclat **(stdin)**. També hi ha comandes que ens donen la seva sortida pel que anomenem sortida estàndard, que és la pantalla **(stdout)**. Tots els errors que pugui produir una comanda es dirigeixen a la sortida d'errors **(stderr)**.

Redireccionament d'E/S:
El redireccionament d'E/S ens permetrà que prenem les dades d'entrada, sortida i error i les redireccionem cap a un altre fitxer diferents als donats per defecte.

j@j:~$ echo "aixo es el contingut del fitxer 1" > fit1 j@j:~$ cat fit1 fit2 aixo es el contingut del fitxer 1 cat: fit2: No such file or directory j@j:~$ cat fit1 fit2 >> copiafit 2>> error.log j@j:~$ cat error.log cat: fit2: No such file or directory	creem un fitxer fit1 fit1 existeix, però fit2 no afegim fit1 al fitxer copiafit i l'error anirà a error.log

j@j:~$ cat fit1 fit2 2> fit3 1>&2 j@j:~$ cat fit1 fit2 > fit3 2>&1 j@j:~$ cat fit3 aixo es el contingut del fitxer 1 cat: fit2: No such file or directory	totes les sortides estan a fit3 perquè els errors van a fit3 i redirigim stdout a stderr totes les sortides estan a fit3 perquè stdout va a fit3 i redirigim stderr a stdout

j@j:~$ cat fit1 fit2 >& fit3 j@j:~$ cat fit3 aixo es el contingut del fitxer 1 cat: fit2: No such file or directory	totes les sortides estan a fit3 perquè redirigim stdout i stderr a fit3

1.6. Filtres

Filtre: És un programa que rep dades per stdin i treu dades per stdout, sense modificar les dades entrades per stdin.

1. Creamos el archivo

j@j:~$ for i in {1..20}; do echo "nose $i" >> readme.txt; done

j@j:~$ cat readme.txt nose 1 nose 2 nose 3 nose 4 nose 5 nose 6 nose 7 nose 8 nose 9 nose 10 nose 11 nose 12 nose 13 nose 14 nose 15 nose 16 nose 17 nose 18 nose 19 nose 20	Mostrem el contingut de tot el fitxer.
j@j:~$ head -3 readme.txt nose 1 nose 2 nose 3	Mostrem les 3 primeres línies.
j@j:~$ tail -f readme.txt nose 11 nose 12 nose 13 nose 14 nose 15 nose 16 nose 17 nose 18 nose 19 nose 20	Mostrem les últimes línies a mida que va canviant.
j@j:~$ wc -l readme.txt 20 readme.txt	Mostrem el número de línies.

j@j:~$ cut -c1-3 /etc/passwd roo dae bin sys syn gam man lp: mai new uuc pro www bac lis irc	Del fitxer /etc/passwd mostra les tres primeres lletres dels usuaris del sistema
j@j:~$ cut -d: -f1,6 /etc/passwd root:/root daemon:/usr/sbin bin:/bin sys:/dev sync:/bin games:/usr/games man:/var/cache/man lp:/var/spool/lpd mail:/var/mail news:/var/spool/news uucp:/var/spool/uucp proxy:/bin www-data:/var/www backup:/var/backups list:/var/list irc:/run/ircd	Del fitxer /etc/passwd mostra el nom i el directori de treball dels usuaris

j@j:~$ grep -i ^m /etc/passwd man:x:6:12:man:/var/cache/man:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin messagebus:x:102:105::/nonexistent:/usr/sbin/nologin j@j:~$ grep ^.*:x:1...: /etc/passwd joel10olor:x:1000:1000:Joel Olivera Organvidez,,,:/home/joel10olor:/bin/bash j@j:~$ grep :/bin/bash$ /etc/passwd root:x:0:0:root:/root:/bin/bash joel10olor:x:1000:1000:Joel Olivera Organvidez,,,:/home/joel10olor:/bin/bash j@j:~$ grep -v :/bin/bash$ /etc/passwd daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin news:x:9:9:news:/var/spool/news:/usr/sbin/nologin uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin proxy:x:13:13:proxy:/bin:/usr/sbin/nologin www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin backup:x:34:34:backup:/var/backups:/usr/sbin/nologin list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin	Mostrar els usuaris que el seu nom comenci per m o M. Mostrar els usuaris que el seu id estigui entre el 1000 i el 1999. Mostrar els usuaris que tinguin per shell /bin/bash Mostrar els usuaris que tinguin qualsevol altre shell.

j@j:~$ cat /etc/passwd | tr ":" "-" root-x-0-0-root-/root-/bin/bash daemon-x-1-1-daemon-/usr/sbin-/usr/sbin/nologin bin-x-2-2-bin-/bin-/usr/sbin/nologin sys-x-3-3-sys-/dev-/usr/sbin/nologin sync-x-4-65534-sync-/bin-/bin/sync games-x-5-60-games-/usr/games-/usr/sbin/nologin man-x-6-12-man-/var/cache/man-/usr/sbin/nologin lp-x-7-7-lp-/var/spool/lpd-/usr/sbin/nologin mail-x-8-8-mail-/var/mail-/usr/sbin/nologin	Mostra per pantalla el contingut del fitxer /etc/passwd amb els camps separats per un -.
j@j:~$ cat /etc/passwd | tr -s "," root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin	Mostra per pantalla el contingut del fitxer /etc/passwd reduint totes les comes a una.
j@j:~$ sed 's/\/bin\/bash/\/bin\/sh/' /etc/passwd root:x:0:0:root:/root:/bin/sh daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin	Mostra per pantalla el contingut del fitxer /etc/passwd canviant la shell /bin/bash per /bin/sh.
j@j:~$ sed 's/\/bin\/bash/\/bin\/sh/' /etc/passwd root:x:0:0:root:/root:/bin/sh daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin mail:x:8:8:mail:/var/mail:/usr/sbin/nologin	Mostra per pantalla el contingut del fitxer /etc/passwd ordenat pel nom d'usuari.
j@j:~$ sort -nt: -k3 /etc/passwd root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin bin:x:2:2:bin:/bin:/usr/sbin/nologin sys:x:3:3:sys:/dev:/usr/sbin/nologin sync:x:4:65534:sync:/bin:/bin/sync games:x:5:60:games:/usr/games:/usr/sbin/nologin man:x:6:12:man:/var/cache/man:/usr/sbin/nologin	Mostra per pantalla el contingut del fitxer ordenat per l'identificador d'usuari.

1.7. Canonades (pipelines)

Canonada: Ens permetran redireccionar la sortida d'una comanda com a entrada d'una altra comanda.

```
j@j:~$ cat /etc/passwd | cut -f1,7 -d: | sort | tr ":" "\t"
_apt     /usr/sbin/nologin
avahi-autoipd  /usr/sbin/nologin
avahi    /usr/sbin/nologin
backup/usr/sbin/nologin
bin      /usr/sbin/nologin
colord  /usr/sbin/nologin

j@j:~$ wc -l /etc/profile /etc/hosts | tail -l | cut -f2 -d" "
27
13
40

j@j:~$ cat /etc/passwd | grep ^.*:.*:.*:`id -g`: | cut -f1,7,6 -d: >
grupmeu
j@j:~$ cat grupmeu
joel10olor:/home/joel10olor:/bin/bash

j@j:~$ ps -ef | grep "^root " | tr -s " " | cut -f2,8 -d" " >
procsroot
j@j:~$ cat procsroot
1 /sbin/init
2 [kthreadd]
3 [rcu_gp]
4 [rcu_par_gp]
5 [netns]
7 [kworker/0:0H-events_highpri]
9 [mm_percpu_wq]
10 [rcu_tasks_rude_]
11 [rcu_tasks_trace]

j@j:~$ ls -l /etc | tr -s " " | cut -f3,4 -d" " | sort | uniq > propsetc
j@j:~$ cat propsetc

root dip
root lp
root root
root shadow
```

Mostrar els noms del*s usuari*s donats d'alta en el sistema i la seva shell d'inici, ordenats alfabèticament i separant els dos camps per un tabulador.

Visualitzar un número que sigui la suma de les línies dels fitxers /etc/profile i /etc/hosts.

A partir del fitxer /etc/passwd crear un fitxer anomenat grupmeu que estigui format pel nom d'usuari, shell inicial i directori inicial del*s usuari*s del teu mateix grup.

Obtenir un fitxer anomenat procsroot que contingui l'identificador (PID) i el nom (CMD) de tots els processos que pertanyen a l'usuari* root.

Obtenir un fitxer anomenat propsetc que contingui una llista (sense repeticions) amb els noms del*s propietari*s i grup al*s quals pertanyen els fitxers i directoris del directori /etc.

2. Programació de shell scripts

2.1. Execució de shell scripts

Shell script: No és més que un programa que usa el llenguatge propi de la shell. Hi ha diferents maneres d'executarlo:

- **Exemple 1:**

```
j@j:~$ echo "sleep 222" > dorm.sh
j@j:~$ chmod u+x dorm.sh
j@j:~$ ./dorm.sh
j@j:~$ pstree -p | grep "sleep"
-gnome-terminal-(98602)-+-bash(98636)-+-bash(702609)---sleep(702610)
```

- **Exemple 2:**

```
j@j:~$ echo -e '#!/bin/bash' "\nsleep 222" > dorm.sh
j@j:~$ ./dorm.sh
j@j:~$ pstree -p | grep "sleep"
-bash(703902)---dorm.sh(733307)---sleep(733308)
```

- **Exemple 3:**

```
j@j:~$ . dorm.sh
j@j:~$ pstree -p | grep "sleep"
-gnome-terminal-(98602)-+-bash(98636)-+-sleep(737327)
```

2.2. Depuració de shell scripts

Depuració: Quan fem un programa i els resultats que aquest produeix no són els esperats és molt útil tenir una eina que ens permeti depurarlo. És a dir, una eina que ens permeti fer un seguiment pas a pas de per on va el flux del programa.

```
j@j:~$ cat dorm.sh
SALUDA=”hola”
DESPEDEIX=”adeu”
set x
echo “$SALUDA $LOGNAME”
echo “Usuaris Connectats”
who
set +x
echo “$DESPEDEIX $LOGNAME”
j@j:~$ ./dorm.sh
““hola” joel10olor”
“Usuaris Connectats”
joel10olor tty2         2022-11-05 11:42 (tty2)
““adeu” joel10olor”
```

2.3. Comentaris

Comentaris: Quan programem hem de tenir present que no estem sols al món o, que encara que ho estiguem, la memòria ens pot fallar i no entendre el programa que nosaltres mateixos hem escrit... En resum, que hem de comentar els nostres scripts per fer nos a tots la vida més fàcil :)

```
j@j:~$ cat dorm.sh
#!/bin/bash
SALUDA=”hola”
DESPEDEIX=”adeu”
set x
echo “$SALUDA $LOGNAME”
echo “Usuaris Connectats”
who
set +x
echo “$DESPEDEIX $LOGNAME”
# NO SE QUE COJONES HACE ESTE SCRIPT
```

2.4. Paràmetres i variables especials

Paràmetres: Un paràmetre d'un script seran tots aquells valors que adjuntem quan executem aquest i que volem poder veure dins del programa.

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " pedro;
echo "Hola, $pedro.";
# NO SE QUE COJONES HACE ESTE SCRIPT

j@j:~$ ./dorm.sh
Dime tu nombre papi: profe te quiero
Hola, profe te quiero.
```

Variables especials: Tenim diferents variables definides en qualsevol script que podem usar segons les nostres necessitats.

2.5. Instruccions e/s

Instruccions d'entrada i sortida: Són aquelles instruccions que ens permetran llegir dades de stdin (entrada) i mostrar dades per stdout (sortida).

```
j@j:~$ read varmeva
3
j@j:~$ echo -e "La meva var és:\t\t$varmeva"
La meva var és:                 3
```

2.6. Operadors

Operadors numèrics: són aquells que ens permeten operar amb números i variables que continguin números.

Operadors lògics: són aquells que ens permeten especificar condicions compostes

2.7. Avaluació d'expressions numèriques

```
j@j:~$ expr 3 + 5
8
j@j:~$ ((a=3+5))
j@j:~$ echo $a
8
j@j:~$ a=1
j@j:~$ ((a=$a+1))
j@j:~$ echo $a
2
```

2.8. Especificació de condicions

Condició: Per tal de trencar el flux d'un programa necessitem especificar condicions que ens el bifurquin cap a un cantó o cap a un altre.

CONDICIONS FITXERS	
[-e *fit*]	true si el fitxer existeix
[-d *fit*]	true si el fitxer existeix i és un directori
[-f *fit*]	true si el fitxer existeix i és regular
[-L *fit*]	true si el fitxer existeix i és un enllaç simbòlic
[-r *fit*]	true si el fitxer existeix i té permís de lectura
[-w *fit*]	true si el fitxer existeix i té permís d'escriptura
[-x *fit*]	true si el fitxer existeix i té permís d'execució
[*fit1* -nt *fit2*]	true si fitxer1 és més nou que fitxer2
[*fit1* -ot *fit2*]	true si fitxer1 és més antic que fitxer2
CONDICIONS CADENES	
[*cad*]	true si no és la cadena buida
[-n *cad*]	true si la longitud de *cadena* és diferent a 0
[-z *cad*]	true si la longitud de *cadena* és 0
[*cad1* = *cad2*]	true si *cadena1* i *cadena2* són iguals
[*cad1* != *cad2*]	true si *cadena1* i *cadena2* són diferents
CONDICIONS ENTERS	
[*num1* -eq *num2*]	true si *num1* i *num2* són iguals
[*num1* -ne *num2*]	true si *num1* i *num2* són diferents
[*num1* -gt *num2*]	true si *num1* és més gran que *num2*
[*num1* -ge *num2*]	true si *num1* és més gran o igual que *num2*
[*num1* -lt *num2*]	true si *num1* és més petit que *num2*
[*num1* -le *num2*]	true si *num1* és més petit o igual que *num2*

2.9. Estructures alternatives

Estructures alternatives: Són aquelles que ens permeten executar un tros de codi segons si es compleix o no una condició.

ESTRUCTURA if	ESTRUCTURA case

if *condicio1*
then
  instruccions
elif *condicio2*
then
  instruccions
else
  instruccions
fi
	case *$nom_var* in
	  patro1) *instruccions*;;
	  patro2) *instruccions*;;
	  ...
	  patron) *instruccions*;;
	  **) instruccions*;;
	esac

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " nombre;
if [ $nombre == "renardo" ]
then
        echo "tu hamster";
else
        echo "$nombre un fekas";
fi
j@j:~$ ./dorm.sh
Dime tu nombre papi: ouh
ouh un fekas
j@j:~$ ./dorm.sh
Dime tu nombre papi: renardo
tu hamster
```

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " nombre;
case $nombre in
        renardo) echo "tu hamster";;
        *) echo "$nombre un fekas";;
esac
j@j:~$ ./dorm.sh
Dime tu nombre papi: adw
adw un fekas
j@j:~$ ./dorm.sh
Dime tu nombre papi: renardo
tu hamster
```

2.10. Estructures iteratives

Estructures iteratives: Són aquelles que ens permeten executar diversos cops un tros de codi.

ESTRUCTURA while	ESTRUCTURA until	ESTRUCTURA for

while *condicio*
do
  instruccions
done
	until *condicio*
	do
	  instruccions
	done
		for *nom_var* in *llista_valors*
		do
		  instruccions
		done

Ejemplo while:

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " nombre;
while [ $nombre != "renardo" ]
do
        read -p "Bribón, ese no es tu nombre. Dime tu nombre pa: " nombre;
done
echo "Grande $nombre";
j@j:~$ ./dorm.sh
Dime tu nombre papi: alfredo
Bribón, ese no es tu nombre. Dime tu nombre pa: bustamante
Bribón, ese no es tu nombre. Dime tu nombre pa: renardo
Grande renardo
```

Ejemplo Until:

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " nombre;
until [ $nombre != "renardo" ]
do
        read -p "Bribón, ese no es tu nombre. Dime tu nombre pa: " nombre;
done
echo "Grande $nombre";
j@j:~$ ./dorm.sh
Dime tu nombre papi: renardo
Bribón, ese no es tu nombre. Dime tu nombre pa: renardo
Bribón, ese no es tu nombre. Dime tu nombre pa: awpdkwad
Grande awpdkwad
```

Ejemplo for:

```
j@j:~$ cat dorm.sh
#!/bin/bash
read -p "Dime tu nombre papi: " nombre;
for i in {1..5}
do
        echo "$nombre, vaya tonto";
done
j@j:~$ ./dorm.sh
Dime tu nombre papi: jorge
jorge, vaya tonto
jorge, vaya tonto
jorge, vaya tonto
jorge, vaya tonto
jorge, vaya tonto
```

3. Una mica més…

3.1. taules

Taula: És una estructura de dades composta.

TAULES	
nom_arr=(*val1 val2 ... valn*)	declaració i assignació inicials
declare -a *nom_arr*	declaració d'una taula. Útil per fer assignacions dinàmiques.
nom_arr[*index*]=*val*	assignació de *val* a l'element *index* de *nom_arr*
${*nom_arr*[*index*]}	fer referència a l'element situat a *index* de la taula *nom_arr*

EXPANSIONS	
${*nom_arr*[#]}	número d'elements de la taula
${*nom_arr*[@]}	llista els elements de la taula tractats cadascun d'ells com a una cadena
${*nom_arr*[*]}	llista els elements de la taula tractats com a una única cadena
${#*nom_arr*[*index*]}	longitud de ${*nom_arr*[*index*]}

```
j@j:~$ nosequeponer=(1 2 3 4 5 6 7 8)
j@j:~$ echo ${nosequeponer[0]}
1
j@j:~$ echo ${nosequeponer[1]}
2
j@j:~$ echo ${nosequeponer[2]}
3
```

3.2. Funcions

Definició d'una funció: Per definir una funció tenim dues possibilitats:

```
nom_funcio(){                    function nom_funcio{
    …                                …
    instruccions                     instruccions
    …                                …
}                                }
```

Paràmetres: Per fer referència als paràmetres que ens puguin arribar a la funció ho farem com hem explicat a l'apartat 2.4. ($1, ..., $n,...). Tots els paràmetres es passen per valor, és a dir quan retornem de la funció el valor dels paràmetres no haurà canviat.

Retorn: Podem fer que les funcions retornin un valor.

3.3. Expressions regulars

Expressions regulars: Són una eina per cercar coincidències entre un text i un patró. L'explicació donada aquí està basada en les expressions regulars estil Perl.

3.4. Comandes moooolt útils

3.4.1. sed

sed: és un editor no interactiu, rep l'entrada per stdin o per un fitxer, fa les operacions que pertoquin i treu el resultat per stdout. De totes les seves funcionalitats detallarem aquí tres de les més potents: escriure, esborrar i substituir. La sintaxi és:

sed [opcions] {operador | script fitxer}

OPCIONS	
-r	per poder usar expressions regulars complexes. Per exemple, per poder fer referència a grups: podem utilitzar *\n* per fer referència a l'*n*-èsim grup, on cada grup es delimita per (expr).
-f	si volem usar un script enlloc d'un operador
-n	no escriu per pantalla la línia que està tractant

OPERADORS BÀSICS	
rangp	imprimeix les línies de *rang*
rangd	esborra les línies de *rang*
s/patro1/patro2/[modificador]	substitueix la primera ocurrència de *patro1* per *patro2*
rang/s/patro1/patro2/[modificador]	substitueix la primera ocurrència de *patro1* per *patro2* de les línies de *rang*

MODIFICADORS	
g	substitueix a totes les ocurrències de les línies, no només a la primera
i	no distingeix entre minúscules i majúscules

j@j:~$ for i in {1..20}; do echo "nose $i" >> readme.txt; done

j@j:~$ cat readme.txt nose 1 nose 2 nose 3 nose 4 nose 5 nose 6 nose 7 nose 8 nose 9 nose 10	j@j:~$ sed 8d readme.txt nose 1 nose 2 nose 3 nose 4 nose 5 nose 6 nose 7 nose 9 nose 10

3.4.2. awk

És un llenguatge de programació dissenyat per processar dades basades en text, ja siguin fitxers o fluxos de dades. La invocació de awk es composa d'un programa i d'un o més fitxers. Cadascuna de les accions del programa s'executa per a cada línia del/s fitxer/s.
Un programa awk estar composat per una o diverses estructures amb la següent sintaxi:

expressió { acció1; acció2; ... }

Expressió: expressió lògica o regular
Accions: Si l'avaluació de l'expressió és positiva s'executa la/les acció/ns indicada/es

```
j@j:~$ echo -e "\nun\ndos\ntres" | awk '{ print "Jo uso"; print "Linux"  }'
Jo uso
Linux
Jo uso
Linux
Jo uso
Linux
Jo uso
Linux
```

Plantilles: Enlloc de l'expressió podem posar unes plantilles que ens serveixen perquè s'executi alguna/es acció/ns abans i/o al finalitzar el programa.

BEGIN {accions_inicials}
expressio1 {accions}
expressio2 {accions}
...
END {accions_finals}

Camps: Cada línia del fitxer que li passem a awk està formada per camps (si no li passem un delimitador, aquest serà l'espai en blanc).

Imprimir camps:

```
j@j:~$ echo "un dos tres" | awk '{ print $1,$2}'
un dos
```

Extreure d'un fitxer les línies on apareix la paraula Linux:

```
j@j:~$ for i in {1..3}; do echo -e "linux torvalds es mi padre \nwindows la chupa como nadie"
>> readme.txt; done
j@j:~$ cat readme.txt
linux torvalds es mi padre
windows la chupa como nadie
linux torvalds es mi padre
windows la chupa como nadie
linux torvalds es mi padre
windows la chupa como nadie
j@j:~$ awk '/linux/' readme.txt
linux torvalds es mi padre
linux torvalds es mi padre
linux torvalds es mi padre
```

Posar l'extensió .nou als fitxers que tenen l'extensió .new:

```
j@j:~$ ls
 dead.letter       fit1          Music     readme.txt
 dedsec-grub-theme  fit3         Pictures  saludo.txt
j@j:~$ ls -l *.txt | awk '{print "mv "$9" "$9".nou"}' | bash
j@j:~$ ls
 dead.letter       fit1          Music     readme.txt.nou
 dedsec-grub-theme  fit3         Pictures  saludo.txt.nou
```

Esborrar tots els directoris, però no els fitxers:

```
joel10olor@EmpanadillaDeAtun:~/PRUEBA$ ls
dir1  dir2  dir3
joel10olor@EmpanadillaDeAtun:~/PRUEBA$ ls -l | grep  '^d' | awk '{print "rm -r "$9}' | bash
joel10olor@EmpanadillaDeAtun:~/PRUEBA$ ls -l
total 0
```

Esborrar tots els fitxers, però no els directoris:

```
joel10olor@EmpanadillaDeAtun:~/pruebas$ ls
A  archivo  C  guay
joel10olor@EmpanadillaDeAtun:~/pruebas$ ls -l | grep -v ^d | awk '{print "rm "$9}' | bash
joel10olor@EmpanadillaDeAtun:~/pruebas$ ls
guay
```

Suma de números positius:

```
j@j:~$ cat sumaawk.txt
BEGIN { total=0 }
$1>0{ total=total+$1 }
END {print "El total és",total} i
```

Variables: Com a qualsevol altre llenguatge de programació podem definir variables. No s'han de declarar, en tenim prou assignant hi un valor.

```
j@j:~$ echo -e "Uso el sistema\noperatiu anomenat GNU/Linux" | awk -v OFS="\t" -v ORS="\t" '{print $1, $2, $3}'
Uso el sistema operatiu anomenat GNU/Linux
```

Operadors: Per poder operar tenim els següents operadors:

OPERADORS			
numèrics		**comparació**	
+	suma	>	major que
-	resta	>=	major o igual que
*	multiplicació	<	menor que
/	divisió	>=	menor o igual que
%	residu	==	igual
^		!=	diferent
var++	incrementar var en 1		
var--	decrementar var en 1		
lògics		**concatenació**	
&&	i	**espai**	concatenació
||	o		
!	negació		

Estructures de control: Disposem també de les diferents estructures de control de qualsevol llenguatge de programació. En el cas de awk s'han heretat la sintaxi del llenguatge de programació C.

EST. if	**EST. for**	**EST. while**	**EST. do/while**

if(*cond*) {
instruccions
}
[else {
instruccions
}]
	for (inic;cond;instr){
	instruccions
	}
		while(cond){
		instruccions
		}
			do{
			instruccions
			} *while(cond)*

3.5. una mica de color i moviment...

L'American National Standards Institute (ANSI) proporciona una sèrie de seqüències de caràcters per poder realitzar determinades tasques sota el S.O.

```
joel10olor@EmpanadillaDeAtun:~$ echo -e "\E[1;4;31;42mHola\E[0m"
Hola
```

4. Annexos

4.1. Exercicis

4.1.1 Variables

4.1.1.1. Usant variables

j@j:~$ DIA1="Dilluns" j@j:~$ DIA2="Dimarts" j@j:~$ DIA3="Dimecres" j@j:~$ DIA4="Dijous" j@j:~$ DIA5="Divendres" j@j:~$ DIA6="Dissabte" j@j:~$ DIA7="Diumenge" j@j:~$ echo $DIA{1..7} Dilluns Dimarts Dimecres Dijous Divendres Dissabte Diumenge j@j:~$ echo -e "Setmana:" $DIA{1..7} Setmana: Dilluns Dimarts Dimecres Dijous Divendres Dissabte Diumenge j@j:~$ SETMANA="$DIA1 $DIA2 $DIA3 $DIA4 $DIA5 $DIA6 $DIA7" j@j:~$ echo $SETMANA Dilluns Dimarts Dimecres Dijous Divendres Dissabte Diumenge	Assignar el valor “Dilluns” a la variable DIA1, el valor “Dimarts” a la variable DIA2 i així consecutivament fins a la variable DIA7. Mostrar el valor de totes les variables per verificar. Usant aquestes variables obtenir la sortida: Setmana: Dilluns Dimarts Dimecres Dijous Divendres Dissabte Diumenge Usant només les variables definides, carregar a la variable SETMANA la llista de dies separats per espais.

4.1.1.2. Usant variables d'entorn

j@j:~$ echo $HOME /home/joel10olor j@j:~$ echo $USER joel10olor j@j:~$ echo $HOSTNAME EmpanadillaDeAtun j@j:~$ echo $SHELL /bin/bash j@j:~$ set BASH_ALIASES=() BASH_ARGC=([0]="0") BASH_ARGV=() j@j:~$ env SHELL=/bin/bash SESSION_MANAGER=local/EmpanadillaDeAtun:@/tmp/.ICE-unix/4576,unix/EmpanadillaDeAtun:/tmp/.ICE-unix/4576	camí del directori de treball de l'usuari nom de login de l'usuari nom de la terminal en ús de l'usuari nom de d'intèrpret de comandes actual camí o camins de cerca d'executables totes les variables d'entorn

4.1.1.3. Usant variables

j@j:~$ VIES="/usr/doc:/var/lib/dpkg" j@j:~$ echo $VIES /usr/doc:/var/lib/dpkg j@j:~$ VIES="/usr/DOC/FAQ:$VIES:/usr/doc/HOWTO" j@j:~$ echo $VIES /usr/DOC/FAQ:/usr/doc:/var/lib/dpkg:/usr/doc/HOWTO	Declara la variable VIES amb el valor "/usr/doc:/var/lib/dpkg". Mostra el seu contingut. Agregar a la variable VIES el directori /usr/doc/HOWTO al final i /usr/DOC/FAQ al principi (separats tots els directoris amb :)

4.1.1.4 . Quina és la sortida de les següents comandes?

j@j:~$ echo $LOGNAME joel10olor j@j:~$ echo "$LOGNAME" joel10olor j@j:~$ echo '$LOGNAME' $LOGNAME j@j:~$ echo \"$LOGNAME\" "joel10olor" j@j:~$ echo "El meu login és $LOGNAME" El meu login és joel10olor j@j:~$ echo 'El meu login és $LOGNAME' El meu login és $LOGNAME	echo $LOGNAME echo "$LOGNAME" echo '$LOGNAME' echo \"$LOGNAME\" echo "El meu login és $LOGNAME" echo 'El meu login és $LOGNAME'

4.1.1.5. Usant variables

j@j:~$ SAVE=$PS1 j@j:~$ echo $SAVE \[\e]0;\u@\h: \w\a\]${debian_chroot:+($debian_chroot)} \[\033[01;32m\]\u@\h\[\033[00m\]:\[\033[01;34m\]\w\[\033[00m\]\$ j@j:~$ PS1="UNIX llest$" UNIX llest$ UNIX llest$PS1=$SAVE j@j:~$	Visualitza el contingut de la variable on es guarda el prompt. Guarda el seu valor en una variable local que s'anomeni SAVE. Canviar el prompt per a que es mostri així: UNIX llest$ Fes que el prompt torni a tenir el seu valor inicial usant la variable SAVE.

4.1.1.6. Usant variables

j@j:~$ VAR1="shell vash 1" j@j:~$ echo $VAR1 shell vash 1 j@j:~$ csh % echo $VAR1 VAR1: Undefined variable. j@j:~$ export VAR1 j@j:~$ csh % echo $VAR1 shell vash 1 j@j:~$ bash j@j:~$ echo $VAR1 shell vash 1	Inicialitzar la variable VAR1 amb la cadena “shell bash 1”. Mostrar el seu contingut. Invocar la shell csh. Quin valor té ara la variable VAR1? Per què? Surt de csh. Quin valor té ara VAR1? Per què? Fes que la variable VAR1 sigui una variable d'entorn i repeteix els passos a) i b) Què ha passat? Si en lloc d'invocar csh invoquem bash. Passa el mateix?

4.1.1.7. Afegeix el directori

/usr/prog/bin a la variable PATH. Com ho faries per comprovar que funciona bé el nou PATH?

j@j:~$ PATH="/usr/prog/bin/

> echo PATH

4.1.1.8. Crea la variable MID

que contingui els valors de les variables HOME i LOGNAME separats per dos punts.

j@j:~$ MID="$HOME:$LOGNAME"

j@j:~$ echo $MID

/home/joel10olor:joel10olor

4.1.2 Redireccionament

4.1.2.9. Crea l'arxiu línies

amb el text “Arxiu línies” com a contingut. Agregar a l'arxiu creat una línia de text, per exemple “Aquesta és la línia 1” sense usar cap editor de text.

```
j@j:~$ rm linies
j@j:~$ echo "Arxiu línies" > linies
j@j:~$ echo "Aquesta és la línia 1" >> linies
j@j:~$ cat linies
Arxiu línies
Aquesta és la línia 1
```

Amb la comanda cat, mostra per pantalla el contingut de l'arxiu /etc/services

```
j@j:~$ cat /etc/services
# Network services, Internet style
#
# Updated from
https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numb
ers.xhtml .
#
```

Escriu la comanda cat redireccionant l'entrada estàndard des de l'arxiu /etc/services i la sortida estàndard cap a l'arxiu serveis.txt. Visualitza serveis.txt

```
j@j:~$ cat /etc/services > serveis.txt
j@j:~$ cat serveis.txt | head
# Network services, Internet style
#
# Updated from
https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numb
ers.xhtml .
#
```

Usant echo, crea l'arxiu errors.txt amb el contingut “Arxiu d Errors”

```
j@j:~$ echo "Arxiu d Errors" > errors.txt
```

Amb la comanda cat intenta mostrar l'arxiu noexist.xxx sense redireccionar entrada estàndard, però redireccionant la sortida estàndard cap a noexist.txt i l'error estàndard cap a agregar a l'arxiu errors.txt. Visualitza noexist.txt i errors.txt.

```
j@j:~$ cat noexist.xxx > noexist.txt 2> errors.txt
j@j:~$ cat noexist.txt
j@j:~$ cat errors.txt
cat: noexist.xxx: No such file or directory
```

4.1.2.10. Utilitza la següent sentència
" $ cat f1 f2 ". El fitxer f1 ha d'existir i el f2 no ha d'existir. Observa la sortida pel monitor.
j@j:~$ echo "nose" > f1
j@j:~$ cat f1 f2
nose
cat: f2: No such file or directory

a) Aconsegueix que la sortida d'errors vagi a un fitxer anomenat errors.
j@j:~$ cat f1 f2 2> errors
nose
j@j:~$ cat errors
cat: f2: No such file or directory

b) Aconsegueix que el contingut del fitxer que existeix es copii a un fitxer anomenat f3 (a més del que has aconseguit a l'apartat a)).
j@j:~$ cat f1 errors > f3
j@j:~$ cat f3
nose
cat: f2: No such file or directory

4.1.2.11. Usant redireccionament

crea un fitxer que contingui la següent informació:
Una línia amb el contingut “Informe del sistema”
Dos línies en blanc
El calendari del mes actual
La data actual
El tipus d'ordinadors que estem usant
Els usuaris que estan connectats

```
j@j:~$ echo -e "Informe del sistema \n\n"  > informe.txt
j@j:~$ cal >> informe.txt
j@j:~$ date >> informe.txt
j@j:~$ sudo dmidecode | grep -A 9 "System Information" >> informe.txt
j@j:~$ who >> informe.txt
j@j:~$ cat informe.txt
Informe del sistema


   Noviembre 2022
do lu ma mi ju vi sá
      1  2  3  4  5
 6  7  8  9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

sáb 05 nov 2022 22:15:19 CET
System Information
        Manufacturer: LENOVO
        Product Name: 82SD
        Version: IdeaPad 5 14IAL7
        Serial Number: MP27EBHS
        UUID: 4494af66-fd9b-11ec-8c90-e4a8dfe049be
        Wake-up Type: Power Switch
        SKU Number: LENOVO_MT_82SD_BU_idea_FM_IdeaPad 5 14IAL7
        Family: IdeaPad 5 14IAL7

joel10olor tty2        2022-11-05 11:42 (tty2)
```

4.1.3 Filtres i Pipelines

4.1.3.12. Donat el següent fitxer

anomenat text:

```
a
aa
ab
aba
aaa
abab
abba
La cadena a es capicua
Tambe es capicua la cadena aa
La cadena ab no es capicua
En una cadena capicua el seu final reflexa el seu principi
Si concatenes una cadena i el seu reflex el resultat es capicua
A
AA
ABA
```

a) Llistar les línies que continguin una lletra ‘a’

```
j@j:~$ cat text.txt | grep "a"
a
aa
ab
aba
aaa
abab
abba
La cadena a es capicua
Tambe es capicua la cadena aa
La cadena ab no es capicua
En una cadena capicua el seu final reflexa el seu
Si concatenes una cadena i el seu reflex el resultat es
capicua
```

b) Llistar les línies que continguin dos lletres ‘a’ consecutives

```
j@j:~$ cat text.txt | grep "aa"
aa
aaa
Tambe es capicua la cadena aa
```

c) Llistar les línies que no continguin la lletra ‘a’

```
j@j:~$ cat text.txt | grep -v "a"
principi
A
AA
```

ABA

d) Llistar les línies que no continguin lletres majúscules
j@j:~$ cat text.txt | grep -v '[[:upper:]]'
a
aa
ab
aba
aaa
abab
abba
principi
capicua

e) Llistar les línies que comencin per la lletra ‘a’
j@j:~$ cat text.txt | grep -v ^a | grep -v ^A
La cadena a es capicua
Tambe es capicua la cadena aa
La cadena ab no es capicua
En una cadena capicua el seu final reflexa el seu
principi
Si concatenes una cadena i el seu reflex el resultat es
capicua

f) Llistar les línies que comencin per una lletra minúscula
j@j:~$ cat text.txt | grep -v '[[:upper:]]'
a
aa
ab
aba
aaa
abab
abba
principi
capicua

g) Llistar les línies que acabin amb la cadena ‘capicua’
j@j:~$ cat text.txt | grep -v capicua$
a
aa
ab
aba
aaa
abab
abba
Tambe es capicua la cadena aa
En una cadena capicua el seu final reflexa el seu

principi
Si concatenes una cadena i el seu reflex el resultat es
A
AA
ABA

4.1.3.13. Llistar tots els arxius

del directori actual sense que apareguin els directoris.

```
j@j:~$ ls -l | grep -v "d"
total 204
-rw-rw-r--  1 joel10olor joel10olor    0 nov  5 18:34 72
-rw-rw-r--  1 joel10olor joel10olor   34 nov  5 22:29 capicupa.txt
-rw-rw-r--  1 joel10olor joel10olor   34 nov  5 12:56 copiafit
-rw-rw-r--  1 joel10olor joel10olor   37 nov  5 12:56 error.log
-rw-rw-r--  1 joel10olor joel10olor   35 nov  5 20:42 errors
-rw-rw-r--  1 joel10olor joel10olor   44 nov  5 20:39 errors.txt
-rw-rw-r--  1 joel10olor joel10olor  212 oct  7 16:13 estructura.xml
-rw-rw-r--  1 joel10olor joel10olor    5 nov  5 20:42 f1
-rw-rw-r--  1 joel10olor joel10olor   40 nov  5 20:45 f3
-rw-rw-r--  1 joel10olor joel10olor   34 nov  5 12:56 fit1
-rw-rw-r--  1 joel10olor joel10olor   71 nov  5 13:01 fit3
-rw-rw-r--  1 joel10olor joel10olor    0 nov  5 16:10 gener
-rw-rw-r--  1 joel10olor joel10olor   38 nov  5 16:06 grupmeu
-rw-rw-r--  1 joel10olor joel10olor  554 nov  5 22:21 informe.txt
-rw-rw-r--  1 joel10olor joel10olor    0 nov  5 16:10 june
-rw-rw-r--  1 joel10olor joel10olor   38 nov  5 20:35 linies
-rw-rw-r--  1 joel10olor joel10olor   16 nov  5 12:53 mailpr
-rw-rw-r--  1 joel10olor joel10olor    0 nov  5 20:39 noexist.txt
-rw-rw-r--  1 joel10olor joel10olor 5576 nov  5 16:06 procsroot
-rw-rw-r--  1 joel10olor joel10olor   40 nov  5 16:07 propsetc
-rw-rw-r--  1 joel10olor joel10olor 12813 nov  5 20:36 serveis.txt
-rw-rw-r--  1 joel10olor joel10olor   76 nov  5 18:28 sumaawk.txt
-rw-rw-r--  1 joel10olor joel10olor  238 nov  5 23:08 text.txt
```

4.1.3.14. Mostrar tots els processos

del sistema amb UID root.

```
j@j:~$ top | grep "root"
    1 root      20   0  168436  13684   8072 S   0,0   0,1   0:06.01 systemd
    2 root      20   0       0      0      0 S   0,0   0,0   0:00.09 kthreadd
    3 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_gp
    4 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 rcu_par_gp
    5 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 netns
    7 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/0:0H+
    9 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 mm_percpu_wq
   10 root      20   0       0      0      0 S   0,0   0,0   0:00.00 rcu_tasks_ru+
   11 root      20   0       0      0      0 S   0,0   0,0   0:00.00 rcu_tasks_tr+
   12 root      20   0       0      0      0 S   0,0   0,0   0:00.50 ksoftirqd/0
   13 root      20   0       0      0      0 I   0,0   0,0   1:13.84 rcu_sched
   14 root      rt   0       0      0      0 S   0,0   0,0   0:00.22 migration/0
   15 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 idle_inject/0
   17 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/0
   18 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/1
   19 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 idle_inject/1
   20 root      rt   0       0      0      0 S   0,0   0,0   0:00.72 migration/1
   21 root      20   0       0      0      0 S   0,0   0,0   0:00.09 ksoftirqd/1
   23 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/1:0H+
   24 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/2
   25 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 idle_inject/2
   26 root      rt   0       0      0      0 S   0,0   0,0   0:00.29 migration/2
   27 root      20   0       0      0      0 S   0,0   0,0   0:00.45 ksoftirqd/2
   29 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/2:0H+
   30 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/3
   31 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 idle_inject/3
   32 root      rt   0       0      0      0 S   0,0   0,0   0:00.52 migration/3
   33 root      20   0       0      0      0 S   0,0   0,0   0:00.01 ksoftirqd/3
   35 root       0 -20       0      0      0 I   0,0   0,0   0:00.00 kworker/3:0H+
   36 root      20   0       0      0      0 S   0,0   0,0   0:00.00 cpuhp/4
   37 root     -51   0       0      0      0 S   0,0   0,0   0:00.00 idle_inject/4
   38 root      rt   0       0      0      0 S   0,0   0,0   0:00.28 migration/4
```

4.1.3.15. Mostrar tots

els processos del sistema amb UID diferent de root.

```
top - 23:19:19 up 11:37,  1 user,  load average: 0,60, 0,73, 0,58
Tasks: 349 total,   1 running, 348 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,7 us,  0,4 sy,  0,0 ni, 98,9 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem :  15724,7 total,   7862,6 free,   2187,0 used,   5675,1 buff/cache
MiB Swap:   9910,3 total,   9910,3 free,      0,0 used.  12777,0 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
 4590 joel10o+  20   0 6249568 329836 122100 S   6,2   2,0  27:27.67 gnome-shell
```

4.1.3.16. Extreure els noms

dels usuaris del resultat de l'ordre “who”.

```
j@j:~$ who | cut -c1-10
joel10olor
```

4.1.3.17. Extreure els camps 1 i 3

del resultat de l'ordre “who”.

```
j@j:~$ who | cut -f1,3
joel10olor tty2         2022-11-05 11:42 (tty2)
```

4.1.3.18. Extreure els permisos

de tots els fitxers del directori $HOME.

```
j@j:~$ ls -l | cut -c12-100

 1 joel10olor joel10olor     0 nov  5 18:34 72
 2 joel10olor joel10olor  4096 sep  9 21:22 8zugg22g.default
10 joel10olor joel10olor  4096 sep  7 13:00 adapta-gtk-theme
 6 joel10olor joel10olor  4096 oct 28 16:40 backup
 1 joel10olor joel10olor    34 nov  5 22:29 capicupa.txt
 6 joel10olor joel10olor  4096 oct 28 16:25 castelar
 1 joel10olor joel10olor    34 nov  5 12:56 copiafit
 2 joel10olor joel10olor  4096 oct  1 18:36 correo
 1 joel10olor joel10olor   193 nov  5 12:54 dead.letter
 5 joel10olor joel10olor  4096 sep  7 00:51 dedsec-grub-theme
 2 joel10olor joel10olor  4096 sep  7 00:29 Desktop
 3 joel10olor joel10olor  4096 oct 27 20:24 Documents
 1 joel10olor joel10olor   105 nov  5 17:22 dorm.sh
 7 joel10olor joel10olor 12288 nov  5 11:47 Downloads
 1 joel10olor joel10olor    37 nov  5 12:56 error.log
 1 joel10olor joel10olor    35 nov  5 20:42 errors
 1 joel10olor joel10olor    44 nov  5 20:39 errors.txt
 1 joel10olor joel10olor   212 oct  7 16:13 estructura.xml
 1 joel10olor joel10olor     5 nov  5 20:42 f1
 1 joel10olor joel10olor    40 nov  5 20:45 f3
```

4.1.3.19. Llistar el propietari

i mida de tots els fitxers del directori $HOME.

```
j@j:~$ ls -l | head | cut -c15-42

joel10olor joel10olor     0
joel10olor joel10olor  4096
joel10olor joel10olor  4096
joel10olor joel10olor  4096
joel10olor joel10olor    34
joel10olor joel10olor  4096
joel10olor joel10olor    34
joel10olor joel10olor  4096
```

4.1.3.20. El fitxer /etc/passwd

del servidor Linux conté informació de tots els usuaris que tenen un compte a la màquina. Cada línia correspon a un usuari diferent i en ella apareixen els següents camps delimitats per ":"

Nom d'usuari
Password codificada en forma de x
Identificador de l'usuari
Identificador del grup al qual pertany l'usuari
Informació respecte l'usuari (nom, cognoms, etc.)
Directori de treball Intèrpret de comandes utilitzat

Un exemple de la línia corresponent a un usuari en aquest fitxer seria:
maria:x:210:204:Maria Sanchez:/home/maria:/bin/bash

a) Comprova si existeix o no una entrada del teu usuari en aquest fitxer.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | grep "joel10olor"
joel10olor:x:1000:1000:Joel Olivera Organvidez,,,:/home/joel10olor:/bin/bash

b) Llista tots els usuaris del mateix grup que tu que existeixin en aquest fitxer.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | grep "1000"
joel10olor:x:1000:1000:Joel Olivera Organvidez,,,:/home/joel10olor:/bin/bash

c) Llista els identificadors d'usuari.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | cut -d ":" -f 3
0
1
2
3
4
5
6
7
8
9
10
13
33
34

d) Llista els shells usats pels usuaris que tinguin el mateix grup que tu.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | grep "1000" | cut -d ":" -f 1
joel10olor

e) Llista els camps 1 i els camps 3 a 5.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | cut -d ":" -f 1,3,4,5
root:0:0:root
daemon:1:1:daemon

bin:2:2:bin
sys:3:3:sys
sync:4:65534:sync
games:5:60:games
man:6:12:man
lp:7:7:lp
mail:8:8:mail
news:9:9:news

4.1.3.21. Mostrar el contingut

del fitxer /etc/passwd ordenat alfabèticament.

```
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | sort -n | head
_apt:x:105:65534::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:110:119:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
avahi:x:114:121:Avahi mDNS daemon,,,:/run/avahi-daemon:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
colord:x:122:129:colord colour management daemon,,,:/var/lib/colord:/usr/sbin/nologin
cups-pk-helper:x:115:122:user for cups-pk-helper
service,,,:/home/cups-pk-helper:/usr/sbin/nologin
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
dnsmasq:x:112:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
fwupd-refresh:x:128:137:fwupd-refresh user,,,:/run/systemd:/usr/sbin/nologin
```

4.1.3.22. Ordenar alfabèticament

el resultat de l'ordre “who”

```
joel10olor@EmpanadillaDeAtun:~$ who | sort -n
joel10olor tty2         2022-11-05 11:42 (tty2)
```

4.1.3.23. Extreure els noms

dels usuaris del resultat de l'ordre “who” i ordenarlos alfabèticament.

```
joel10olor@EmpanadillaDeAtun:~$ who | cut -c1-10 | sort -n
joel10olor
```

4.1.3.24. Extreure els noms

dels usuaris del resultat de l'ordre “who”, ordenarlos alfabèticament i eliminar els noms repetits.

```
joel10olor@EmpanadillaDeAtun:~$ who | cut -c1-10 | sort -n | uniq -u
joel10olor
```

4.1.3.25. Llistar les mides

i els noms dels fitxers (excloent els directoris) del directori actual ordenats per mida.
joel10olor@EmpanadillaDeAtun:~$ ls -l | grep -v "d" | cut -c37-42,55-70 | sort -n

```
    0 72
    0 gener
    0 june
    0 noexist.txt
    5 f1
   16 mailpr
   34 capicupa.txt
   34 copiafit
   34 fit1
   35 errors
   37 error.log
   38 grupmeu
   38 linies
   40 f3
   40 propsetc
   44 errors.txt
   71 fit3
   76 sumaawk.txt
  212 estructura.xml
  238 text.txt
  554 informe.txt
 5576 procsroot
12813 serveis.txt
```

4.1.3.26. En el fitxer /etc/group

s'especifiquen els grups existents al sistema amb el seu identificador i els seus components. El format de cada línia és el següent:
nom_grup:id_grup:comp1,comp2,...compn

a) Fes sortir per pantalla les línies dels grups on el teu usuari està.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/group | grep "joel10olor"
adm:x:4:syslog,joel10olor
cdrom:x:24:joel10olor
sudo:x:27:joel10olor
dip:x:30:joel10olor
plugdev:x:46:joel10olor
lpadmin:x:122:joel10olor
lxd:x:134:joel10olor
joel10olor:x:1000:
sambashare:x:135:joel10olor

b) Fes sortir per pantalla a quants grups estàs.

joel10olor@EmpanadillaDeAtun:~$ cat /etc/group | grep -c "joel10olor"
9
c) Fes sortir per pantalla les línies dels grups on estàs tu i algun usuari més.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/group | cut -d ":" -f4 | grep -v -n -e '^$'
5:syslog,joel10olor
18:joel10olor
21:joel10olor
22:pulse
23:joel10olor
35:joel10olor
61:joel10olor
66:saned
73:joel10olor
75:joel10olor

d) Fes ara que només surtin els noms dels grups on està el teu usuari.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/group | grep "joel10olor" | cut -d ":" -f1
adm
cdrom
sudo
dip
plugdev
lpadmin
lxd
joel10olor
sambashare

e) Emmagatzema aquests noms de grups en una variable que es digui MEUSGRUPS
joel10olor@EmpanadillaDeAtun:~$ MEUSGRUPS=$(cat /etc/group | grep "joel10olor" | cut -d ":" -f1)
joel10olor@EmpanadillaDeAtun:~$ echo $MEUSGRUPS
adm cdrom sudo dip plugdev lpadmin lxd joel10olor sambashare

f) Fes sortir per pantalla el primer i tercer camp (nom i identificador) de la línia corresponent al teu usuari a l'arxiu /etc/passwd. El nom i identificador estaran separats per ":".
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | grep "joel10olor" | cut -d ":" -f1,3
joel10olor:1000

g) Fes sortir el mateix resultat que a l'apartat f) però ara el nom i l'identificador estaran separats per un espai en blanc.
joel10olor@EmpanadillaDeAtun:~$ cat /etc/passwd | grep "joel10olor" | cut -d ":" -f1,3 | sed "s/:/ /g"
joel10olor 1000

h) Assigna la sortida anterior a una variable anomenada IDENTITAT.
joel10olor@EmpanadillaDeAtun:~$ IDENTITAT=$(cat /etc/passwd | grep "joel10olor" | cut -d ":" -f1,3 | sed "s/:/ /g")
joel10olor@EmpanadillaDeAtun:~$ echo $IDENTITAT
joel10olor 1000

i) Crea una variable que es digui JO amb el contingut de la variable IDENTITAT i de la variable MEUSGRUPS, separats amb un $. Mostra el resultat per pantalla.
joel10olor@EmpanadillaDeAtun:~$ JO="$IDENTITAT $ $MEUSGRUPS"
joel10olor@EmpanadillaDeAtun:~$ echo $JO
joel10olor 1000 $ adm cdrom sudo dip plugdev lpadmin lxd joel10olor sambashare

27. Volem imprimir per pantalla “Hola <num_id>” és el teu identificador. Fesho de tres maneres diferents:
a) Usant la comanda id amb opcions
joel10olor@EmpanadillaDeAtun:~$ echo "Hola" $(id -u)
Hola 1000

b) Usant la comanda id sense opcions
joel10olor@EmpanadillaDeAtun:~$ echo "Hola" $(id | cut -d "=" -f2 | cut -d "(" -f1)
Hola 1000

c) Usant l'arxiu /etc/passwd
joel10olor@EmpanadillaDeAtun:~$ echo "Hola" $(cat /etc/passwd | grep "joel10olor" | cut -d ":" -f3)
Hola 1000

28. Usant la comanda date sense cap opció fes que:
a) Surti el dia (núm) per pantalla. Emmagatzemar el mes a la variable DIA.
joel10olor@EmpanadillaDeAtun:~$ date | cut -c1-3
joel10olor@EmpanadillaDeAtun:~$ date +"%d"
06
joel10olor@EmpanadillaDeAtun:~$ DIA=$(date +"%d")
joel10olor@EmpanadillaDeAtun:~$ echo $DIA
06

b) Surti el mes per pantalla. Emmagatzemar el mes a la variable MES.
joel10olor@EmpanadillaDeAtun:~$ date +"%B"
noviembre
joel10olor@EmpanadillaDeAtun:~$ MES=$(date +"%B")
joel10olor@EmpanadillaDeAtun:~$ echo $MES
noviembre

c) Surti l'any per pantalla. Emmagatzemar el mes a la variable ANY.
joel10olor@EmpanadillaDeAtun:~$ date +"%Y"
2022
joel10olor@EmpanadillaDeAtun:~$ ANY=$(date +"%Y")
joel10olor@EmpanadillaDeAtun:~$ echo $ANY
2022

d) Surti per pantalla “Avui és dia del mes de l'any ”, usant les variables creades.
joel10olor@EmpanadillaDeAtun:~$ echo "Avui és $DIA de $MES de $ANY"
Avui és 06 de noviembre de 2022

e) Guardar la línia anterior en el fitxer avui.dat sense que la sortida es vegi per pantalla.
joel10olor@EmpanadillaDeAtun:~$ echo "Avui és $DIA de $MES de $ANY" > avui.dat

f) Guardar la línia anterior en el fitxer avui.dat i veientse la sortida per pantalla.
joel10olor@EmpanadillaDeAtun:~$ echo "Avui és $DIA de $MES de $ANY" 2> avui.dat | tee avui.dat
Avui és 06 de noviembre de 2022
joel10olor@EmpanadillaDeAtun:~$ cat avui.dat
Avui és 06 de noviembre de 2022

4.1.4. Shell-Scripts

4.1.4.29. Dissenyar un script

que donats dos arguments els sumi si el primer és menor que el segon i els resti en cas contrari.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
read -p "num1: " n1;
read -p "num2: " n2;

if [[ $n1 < $n2 ]]
then
        expr $n1 + $n2
else
        expr $n1 - $n2
fi
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
num1: 1
num2: 2
3
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
num1: 2
num2: 1
1
```

4.1.4.30. Dissenyar un script

que demani un caràcter i ens digui si és un número, una lletra o una altra cosa.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
read -p "Dime un carácter: " n1;

if [[ $n1 =~ ^[0-9]+$ ]]
then
        echo "Es un número";
elif [[ $n1 =~ ^[a-zA-Z]+$ ]]
then
        echo "Es una letra";
else
        echo "No se que es lo que me has pasado crack";
fi

joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
Dime un carácter: 1
Es un número
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
Dime un carácter: a
Es una letra
```

```
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
Dime un carácter: ?
No se que es lo que me has pasado crack
```

4.1.4.31. Dissenyar un script

al qual se li passa un argument i si aquest és un directori llista el seu contingut.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
read -p "Dime un directorio: " n1;

if [[ "$(ls)" == *"$n1"* ]]; then
        ls $n1;
else
        echo "No existe crack";
fi
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
Dime un directorio: aoijdoiajda2dq
No existe crack
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
Dime un directorio: pruebas
guay
joel10olor@EmpanadillaDeAtun:~$ ls pruebas
guay
```

4.1.4.32. Dissenyar un script

que suma tots els números que se li passen per paràmetre.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
for i do
        suma=$(expr $suma + $i)
done
echo $suma
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh 1 2 3
6
```

4.1.4.33. Dissenyar un script

que compti el número de caràcters que tenen els noms dels fitxers del directori actual.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
echo $(ls -l | grep -v "d" | cut -c56-70 | wc -w)
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh
26
```

4.1.4.34. Dissenyar un script

que determini si els números que se li passen com a paràmetres són parells o senars.

```
joel10olor@EmpanadillaDeAtun:~$ cat dorm.sh
#!/bin/bash
for i do
        mod=$(($i % 2))
        if [[ mod -eq 0 ]]
        then
                echo "$i es par"
        else
                echo "$i es impar"
        fi
done
joel10olor@EmpanadillaDeAtun:~$ ./dorm.sh 1 2 3 4
1 es impar
2 es par
3 es impar
4 es par
```

4.1.4.35. Dissenyar un script

que demani un nom d'usuari i ens digui si aquest existeix i, si existeix, ens digui si està conectat.

4.1.4.36. Dissenyar un script

anomenat “lse” que et mostri per pantalla els fitxers i directoris que hi ha al directori actual de la següent manera:

Si amb la comanda ls obtenim la següent sortida

```
fit1  prog1  prog2  joc
```

Amb el nostre script lse ens ho mostrarà així:

```
|_fit1
.|_prog1
..|_prog2
…|_joc
```

4.1.4.37. Dissenyar un script

anomenat “sumatam” que sumi les mides de tots els fitxers que se li passin com a arguments donant un error per a tots aquells arguments que no existeixin o que siguin directoris.

4.1.4.38. Dissenyar un script

anomenat “sumadir” que sumi les mides de tots els fitxers del directori passat com a argument, donant un missatge d'error si l'argument passat no és un directori o no existeix.

39. Dissenyar un script anomenat “opf” que permeti copiar (c), moure (m) i esborrar (d) fitxers. L'script ha de comprovar que la sintaxi utilitzada és la correcta.

4.1.4.40. Dissenyar un script

anomenat “usua” que ens vagi demanant noms d'usuari i si aquests existeixen i estan connectats ens dirà la data i hora de la seva connexió. L'script s'acabarà quan l'usuari teclegi FI.

4.1.4.41. Dissenyar un script

anomenat “lro” que mostri tots els fitxers amb permís de lectura per a altres d'un directori que es proporciona com a paràmetre.