**SCRIPTS BASH**

**nano ej.sh**→editar fichero.

Ctrl+O->Guardar el contenido de fichero

Ctrl+X->Salir del fichero

**cat**→leer archivos, muestra en pantalla.

**touch**→crear archivo.

**#!/usr/bin/python**→ruta de ficheros ejecutables con python

**#!/bin/bash**→principio de fichero.

**bash ej.sh**→ ejecución, desde la carpeta del fichero, si no:

**-chmod** u+x→cambiar permisos y poder ejecutarlo sin encontrarnos en la ruta.

**which**→ muestra ruta de un ejecutable.

which ls → /usr/bin/ls

**file**→muestra tipo de fichero.

file ej1.sh

**echo**→mostrar contenido.

-n→muestra el contenido en la misma línea.

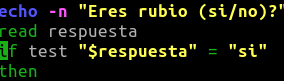
**read**→leer datos de teclado

-p→muestras mensaje y lees variable

read -p "Introduce tu usuario: " usuario

‘ ’

“ ”



Si ponemos la variable entre comillas agrupará la respuesta que le des en un bloque conjunto aunque esta tenga espacios.

**CONDICIONAL:**

**if [ *condición* ] then - else - fi**

**elif**

**test**→ devuelve verdadero/falso

if test $respuesta = “si” then

**man test**→la ayuda del **if** es la ayuda del test (para ver condiciones: mayor, menor…)

**⥤COMPARACIONES NUMÉRICAS**

**-eq**→equal

**-ne**→ not equal

**-gt**→greater than

**-ge**→greater equal

**-lt**→less than

**-le**→less equal

if [ $num1 -gt $num2 ]then…

**⥤COMPARACIONES TEXTO**

**=**

**!=**

**-n**→ dice verdadero si tiene algo escrito

**-z**→ dice verdadero siq está vacío

read -p "Dime tu nombre: " nombre

if [ -z $nombre ] then…

**⥤COMPARACIONES FICHERO**

*Devuelve verdadero:*

**-r**→permiso lectura (se puede leer)

**-w**→permiso escritura

**-x**→permiso de ejecución

**-f**→fichero

**-d**→directorio

**-e**→existe

**-s**→size: te dice si tiene tamaño

if [ -e $fichero ] then…

**NEGACIÓN**

!

**CONJUNCIÓN**

**-a**

**DISYUNCIÓN**

**-o**

if [ *condición -o condición* ] then …

#!/bin/bash

echo -n “Dime tu edad: “

read edad

if [ $edad -eq 18 ] ó [ $edad = 18 ]

then

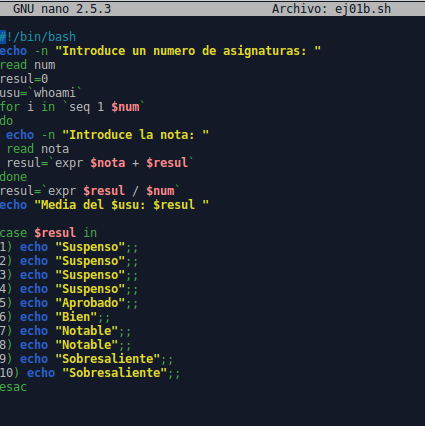
echo “Es legal cabrón”

fi

*con* **-eq** *sí que acepta que metas por ejemplo en este caso 018, un 0 delante. (comparación numérica)*

*con “=” tienen que ser los caracteres exactos (comparación de caracteres)*

**case *variable* in - esac**→es como el switch de javascript



\*) significa que has puesto cualquier otra cosa.

**BUCLES**

**⧭while [ *condición* ] do - done**→mientras que …

**⧭until**→ hasta que…

**sleep**→hacer pausas. sleep 0.5 (segundos)

**seq**→secuencia.

seq 1 4 → 1 2 3 4

seq 1 2 7→ 1 3 5 7

seq 10 -1 1→10 9 8 7 6 … 1

**expr**→resuelve operaciones matemáticas, tienes que dejar espacios entre todos los caracteres, y no te da decimales.

variable=`expr $num + 2`

\\*→ multiplicar

%→módulo (resto de la división entera)

length → cuenta caracteres

variable=`expr length “$cad”`

substr→ trozo de cadena.

variable=`expr substr “cocodrilo” 1 4`

**$RANDOM**→variable números aleatorios.

alea=`expr $RANDOM % 10 (RANGO) + 1 (INICIAL)

**let *variable***→ operaciones **NUMÉRICAS.**

**⧭for variable in conjunto do - done**→

*Ejemplo in:*

*for dia in Lunes “Martes Miércoles” Jueves*

↳OJO!! entre comillas es un mismo elemento.

for in `seq 0 0.5 3`

↳OJO!! comillas invertidas para comandos, sino lo toma como elemento.

***VER EJEMPLOS DEL FOR PQ UFFF…***

**groups**→grupos a los que pertenece el usuario.

**whoami**→usuario actual.

**PARÁMETROS**

Son los datos introducidos al ejecutar un script.

**$0**→ruta del archivo, junto con el archivo.

**$0 basename**→solo el nombre.

**$#**→nos dice el nº de variables introducidas.

**$1**→primera variable introducida, $2, $3… hasta el $9, como máximo.

**$?**→código de salida del último comando ejecutado, si es 0 funciona (verdadero) y si es 1 u otro número falla (falso).

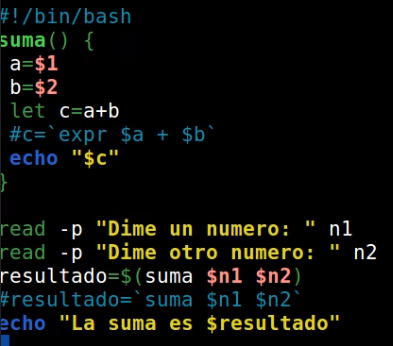
**EJECUCIONES DE COMANDOS Y RECURSOS ADICIONALES**

**`comando a ejecutar`**

**$(comando a ejecutar)**

*No poner $ a la variable si es para guardarle algo.*

* **FUNCIONES:** se define, para poder ejecutar después. Abajo se llama:



* **ARRAYS:**

array1=(“elemento1” “elemento2” “elemento3”...)

${#array1[@]}

# cuenta todos los elementos

@ todos los elementos

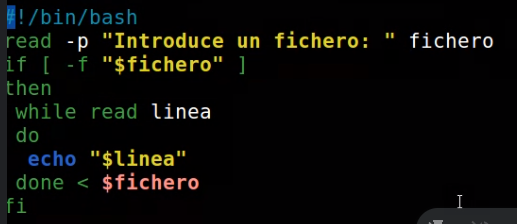
*ej27.sh\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

*¡Cuidado! porque los arrays empiezan en 0 por eso le restamos 1 y empezamos la seq en 0*

* **INTRODUCIR VARIABLES**

while comando

do

done **< variable/documento o > variable/documento**

*ej28.sh\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

* **BUCLE INFINITO**

**while true** (o algo que sea verdadero, ejemplo: 1=1)

**exit**→ sale del script (puede ir a compañado de un numero indicando si ha tenido éxito(**exit 0**) o no(**exit** **1**))

**break**→ salta una ejecución bucle

**date +%s**→cuántos segundos han pasado desde que se creó linux: 1/1/1970

**FILTROS BASH**

**SÍMBOLOS**

**comando1 ; comando2** *↦* separar comandos independientes

**comando1 | comando2** *↦* separar comando dependientes “la información que sale del 1er comando va al 2º”

**comando > fichero** *↦* (comando a fichero, reescribe)

**comando < fichero** *↦* (de fichero a comando, la entrada del comando viene del fichero)

**>>** *↦* (añadir)

**2>** (entrada estándar teclado, salida estándar pantalla; salida de errores)

**2>>**

***/dev/null*** *↦ es un fichero en el que metes cualquier cosa y desaparece/lo borra*

*ej30.sh\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\**

***ctrl + alt + f1*** *↦ distintas terminales máquinas reales*

***ctrl derch + f1*** *↦ distintas terminales máquinas virtuales*

***ctrl derch + f7*** *↦ vuelves a la máquina normal*

**FILTROS**

**head**

⤷Por defecto primeras 10 líneas

⤷ **-n** (nº de líneas)

**tail**

⤷Por defecto últimas 10 líneas

⤷ **-n** (nº de líneas)

**nl**→numera las líneas.

**wc**

⤷ **-l** → contar líneas

⤷ **-w** →contar las palabras

⤷ **-c**→contar caracteres

**more**→mostrar de forma paginada la info.

⤷ **-enter**→línea en línea

⤷ **-espacio**→avanza una página entera

⤷ **-q**→parar

**less**→mostrar de forma paginada la info.

⤷en vez de más pone dos puntos

⤷ **-RePág**→ avanzar

⤷ **-AvPág**→ retroceder

⤷cuando llegas al final no acaba porque puedes retroceder.

**sort**→ordena alfabéticamente

⤷ **-r**→inversamente

⤷ **-R**→orden aleatorio

⤷ **-n**→orden numérico

⤷pones **nl** antes

⤷ **-u**→ordena y quita líneas repetidas (exactamente igual: mins, mayús…)

⤷**uniq**

⤷ **-i**→ no diferencia mayús y mins.

⤷ **-c**→ número de veces que están repetidas (SOLO SI ESTÁN SEGUIDAS)

⤷ **-d**→ sólo aquellas repetidas + de 1 vez.

⤷ **-u**→solo una vez repetida.

⤷**separadores:**

⤷ **-t**→ indica el separador (entre “”)

⤷ **-k**→indica la columna

⤷ **-o**→ lo escribe en un fichero ( -o fichero.txt)

**rev**→muestra el contenido al revés.

**IMPORTANTE EL ORDEN DE LOS FILTROS**

**cut**→quedarnos con un trozo de la línea.

⤷ **-c**→muestra caracteres

cut -c 1-4 (RANGO) || cut -c 2,4,9 (CONCRETOS)

⤷ **-d**→delimitador (entre “”)

⤷ **-f**→ columna (RANGO/CONCRETO) -f 1-4

**tr**→cambia caracteres/conjuntos.

⤷tr **[aeiou]** a

⤷**[:lower:]** minúsculas → **[:upper:]** mayúsculas

*VICEVERSA*

⤷ **-d**→borrar carácteres

echo Goooolll! | tr -d o →→→ Golll!

⤷ **-s** →comprimir

echo Goooolll! | tr -d ol →→→ Gol!

**grep**→filtrados heavys

⤷ **-e**→ muestra todas las líneas donde aparezca la palabra

⤷ **-i** (--ignore-case)→ no diferencia entre mayúsculas y minúsculas

⤷ **-c**(--count)→ muestra el número de líneas que tienen esa palabra (aunque esté repetida en la misma línea)

⤷ **-n**(--line-number)→ muestra las líneas donde aparece y de forma numerada

⤷ **-v** (--invert-match)→ muestra las que no tienen esa palabra

⤷ **-E**→buscas expresiones regulares (patrón)

⤷ **OPCIONES**

⤷ **[aeiou]**→que contenga estas letras.

⤷**[A-Z]**→busca ese rango de letras mayús.

⤷**[0123456789]**→que aparezca num.

⤷**[0-9]**→rango de números === **[[:digit:]]**

⤷**[a-zA-Z]**→letras === **[[:alpha:]]**

⤷**[[:alnum:]]**→números y letras

⤷ **CANTIDAD**

⤷**[]?**→0 o 1 vez

⤷**[]\***→0 o más veces

⤷**[]+**→1 o más veces

⤷**[]{4}**→4 veces

⤷**[]{1-4}**→1 a 4 veces

⤷**[]{-3}**→ hasta 3 veces

⤷**[]{3-}**→ al menos 3 veces

⤷ **CONTRARIO**

⤷**[^aeiou]** ↦ salen consonantes

⤷ **INICIO FRASE**

⤷**^**  grep -E “^coco”

⤷ **FIN FRASE**

⤷**$**

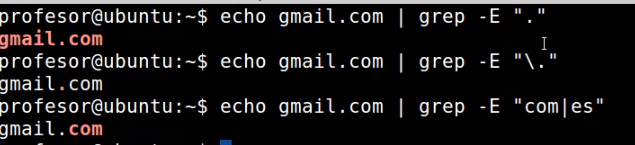
⤷ **CUALQUIER CARÁCTER**

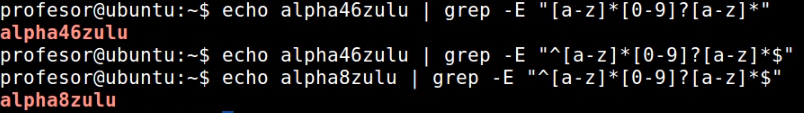
⤷**.**

⤷ **PUNTO**

⤷**\.**

⤷ **IMPORTANTE LAS COMILLAS**







**ls**→ listar directorios

⤷ **-d \*/**→listar sólo directorios

⤷**\***→ listar directorios y subdirectorios

⤷**-R**→recursivamente

⤷**-s**→con tamaño

⤷**-l**→formato largo (permisos/propietarios…)

⤷**-lh**→igual que -l pero con tamaño legible

⤷**-a**→con archivos ocultos

⤷**-1**→aparece en una sola columna

**du**→disk usage

⤷**/**→todo el disco

⤷**-s**→solamente imprime en la salida el uso de espacio para los argumentos realmente dados, no para sus subdirectorios.

⤷**-h** (--human-readable)→añade una letra indicativa del tamaño, como **M** para megabytes binariosa cada tamaño.

⤷**-b** (--bytes)→muestra tamaños en bytes.

**uptime** → indica el tiempo que lleva el

sistema encendido

⤷**-p**→solo horas, minutos y segundos

**poweroff**→ apagar el sistema

⤷**-r**→reinicio en 1 minuto

⤷**-r 0**→reinicio al momento

⤷**-h 0**→ tiempo que quieres que espere hasta apagarse. En este caso sin espera

⤷**-now**→ahora mismo

**mkdir**

**mv**

**users**

**ps**→ listar procesos

⤷ **-e**→los lista con el nombre

⤷ **-ef**→los lista todos los procesos de cad app con ruta