IreneCR2
www.wuolah.com/student/IreneCR2

Práctica 2.pdf

PRÁCTICA 2

2º Programación y Administración de Sistemas

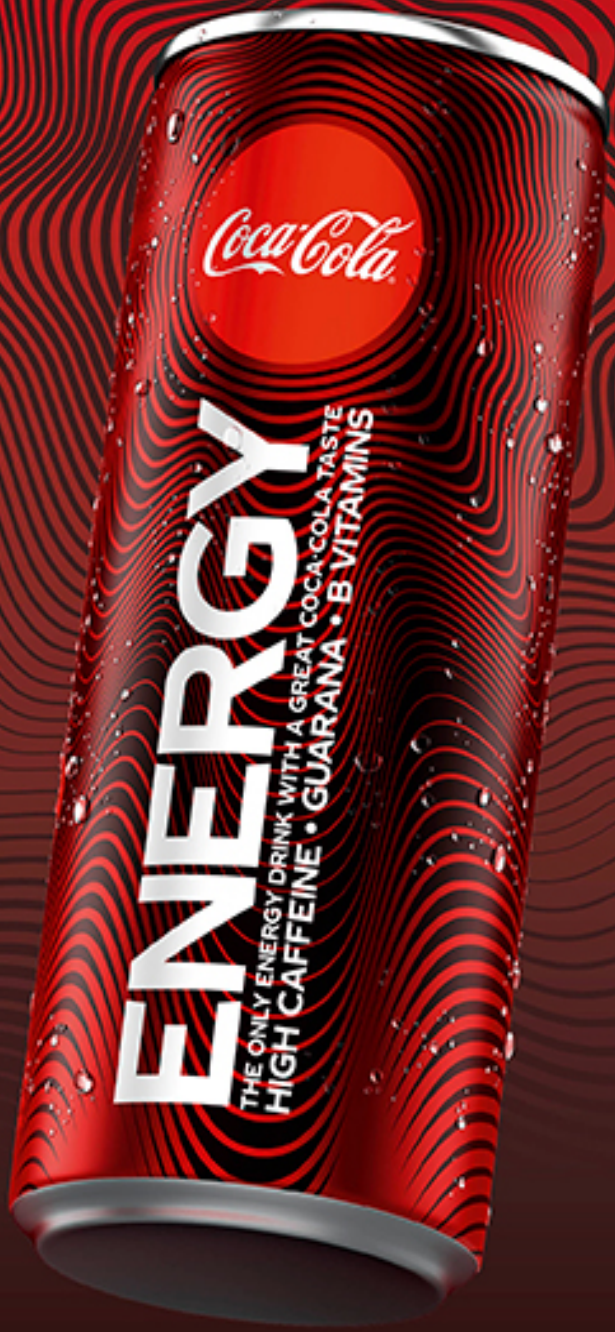
Grado en Ingeniería Informática

Escuela Politécnica Superior de Córdoba
UCO - Universidad de Córdoba

```
# ********** EJERCICIO 1 **********

#Script que recibe el fichero de ejemplo 'peliculas.txt' y ejecuta los comandos adecuados de 'grep' que permiten realizar las siguientes tareas:
     # Mostrar las líneas con la duración de la película (uno o más dígitos al principio de la línea, un espacio y la secuencia "min").
     # Mostrar las líneas que contienen el país de la película (rodeado de guiones, -España-).
     # Mostrar solo los países (sin la línea completa).
     # Contar las películas del 2016 y las del 2017.
     # Mostrar el fichero sin líneas vacías.
     # Mostrar las líneas que empiezan por 'E' (haya o no espacios antes).
     # Mostrar las líneas que contengan una 'd', 'l' o 't', una vocal, y la misma letra de las tres anteriores (Universi'dad').
     # Mostrar las líneas que tengan 8 vocales 'a' o más (mayúsculas o minúsculas y no necesariamente seguidas).
     # Mostrar las líneas que terminan con tres puntos (...) y no empiezan por espacio, utilizando el operador de repetición \{...\} o {...}.
     # Utilizar 'sed' para mostrar, entre comillas dobles, las vocales con tilde (mayúsculas o minúsculas).

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"



# ---------------------------------
#      CONTROL DE ARGUMENTOS
# ---------------------------------

if [ $# -ne 1 ]
then
     echo "ERROR: Debe introducir ./ejercicio1.sh <nombre_fichero.txt>"
     echo -e "\n"
     exit 1
fi

if [ ! -f "$1" ] #Si no es un fichero
then
     echo "$1 no es un fichero"
     echo -e "\n"
     exit 1
fi



# ---------------------------------
#          PROCESAMIENTO
# ---------------------------------
```

LA ÚNICA BEBIDA ENERGÉTICA CON UN GRAN SABOR A COCA-COLA

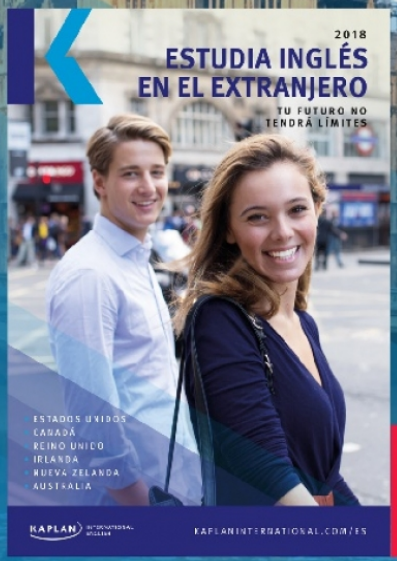
EXPANDE TU ENERGÍA POSITIVA



CURSOS DE INGLÉS EN EL EXTRANJERO

La inversión más inteligente para tu futuro

- 80 años de experiencia en educación internacional.
- 97% de recomendación entre nuestros estudiantes.
- 40 escuelas de inglés acreditadas: Reino Unido, Irlanda, Estados Unidos, Canadá, Australia y Nueva Zelanda.
- Cursos para todos los niveles y objetivos: inglés general, de negocios, preparación de exámenes, larga duración, entre otros.



DESCARGA **EL CATÁLOGO** GRATUITO

KAPLANINTERNATIONAL.COM/ES

KAPLAN INTERNATIONAL ENGLISH

```
echo "************************************************"
echo -e "\n"
echo "1) Líneas con la duración de las películas:"
echo -e "\n"
cat $1 | sed -r -n -e 's/([[:digit:]]*)(hr) ([[:digit:]]*)(min)/\1\2
\3\4/p' #hora'hr' min'min'
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "2) Líneas con el país de las películas:"
echo -e "\n"
#Se buscan las líneas que tienen algo entre guiones, ya que el país viene
entre guiones y se supone que no se utilizarán los guiones para otra cosa.
cat $1 | grep -E '\-.+-'
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "3) Sólo el país de las películas:"
echo -e "\n"
#Igual que en el 2, pero con -o solo seleccionamos la parte que coincide.
Luego quitamos los guiones, quedándonos sólo con el contenido.
cat $1 | grep -o -E '\-.*\-' | grep -o -E '[^-].+[^-]'
echo -e "\n"



echo "************************************************"
echo -e "\n"
#No buscamos solo 2016 y 2017 ya que podría aparecer en el argumento.
Pasamos el resultado del grep a un wc -l para contar las películas de cada
año.
pelis2016=$(cat $1 | grep -E '\([[:digit:]]+/[[:digit:]]+/2016\)' | wc -l)
pelis2017=$(cat $1 | grep -E '\([[:digit:]]+/[[:digit:]]+/2017\)' | wc -l)
echo "4) Hay $pelis2016 películas de 2016 y $pelis2017 películas de 2017"
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "5) Eliminar líneas vacías:"
echo -e "\n"
cat $1 | grep '.' #Muestra todas las líneas en las que haya algo; por
tanto, obvia las vacías. Otra opción sería grep -v '^$', que mostraría
todas las líneas distintas a las vacías (que sean el principio y el final).
Con sed sería sed '/./!d' o también sed '/^$/d'.
```


```
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "6) Líneas que empiezan por la letra E (con o sin espacios antes):"
echo -e "\n"
cat $1 | grep -E '^[[:space:]]*E' #Empieza por ninguno, uno o varios
espacios, seguidos de una E
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "7) Líneas que contienen 'd', 'l' o 't', una vocal y la misma letra:"
echo -e "\n"
cat $1 | grep -E '([dlt])[aeiou]\1'
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "8) Líneas que contienen ocho vocales 'a' o más:"
echo -e "\n"
cat $1 | grep -E -i 'A.*A.*A.*A.*A.*A.*A.*A' #O también '([Aa].*){8,}'
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "9) Líneas que terminan con tres puntos y no empiezan por espacio:"
echo -e "\n"
cat $1 | grep -E '^([^ ]).*(\.{3,3})$'
echo -e "\n"



echo "************************************************"
echo -e "\n"
echo "10) Mostrar entre comillas las vocales con tilde:"
echo -e "\n"
cat $1 | sed -r -e 's/([áéíóúÁÉÍÓÚ])/"\1"/g' #Con la bandera g se aplica a
todo el fichero y no solo a la primera coincidencia.


echo -e "\n"
```

```
# ********** EJERCICIO 2 **********

#Script que recibe el fichero 'peliculas.txt' y realiza las siguientes
operaciones utilizando 'sed':
     # Elimina las líneas vacías y los subrayados (===).
     # Presenta el título de la película como "Título: xxx" (se recomienda
procesarlo al final).
     # Fecha de estreno como "|-> Fecha de estreno: xxx".
     # Director como "|-> Director: xxx".
     # Reparto como "|-> Reparto: xxx".
     # Duración como "|-> Duración: xxx"

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"



# ----------------------------------
#      CONTROL DE ARGUMENTOS
# ----------------------------------

if [ $# -ne 1 ]
then
     echo "ERROR: Debe introducir ./ejercicio2.sh <nombre_fichero.txt>"
     echo -e "\n"
     exit 1
fi

if [ ! -f "$1" ]
then
     echo "$1 no es un fichero."
     exit 1
fi



# ----------------------------------
#          PROCESAMIENTO
# ----------------------------------

cat $1 | sed -e '/^$/d' | sed -r -e '/==+/d' | sed -r -e
's/(\(.*\/.*\/.*\)) (.+)/|-> Fecha de estreno: \1/' | sed -r -e 's/Dirigida
por (.+)$/|-> Director: \1/' | sed -r -e 's/Reparto: (.+)/|-> Reparto: \1/'
| sed -r -e 's/([[:digit:]]hr [[:digit:]]*min)/|-> Duración: \1/' | sed -r
-e '/^ +.+/d' | sed -r -e 's/(^[^(|)].*)/Título: \1/'

echo -e "\n"
```

```
# sed -e '/^$/d'
    # Borra las líneas vacías que hay entre cada película.

# sed -r -e '/==+/d'
    # Borra las líneas que contengan los caracteres '===='

# sed -r -e 's/(\(.*\/.*\/.*\)) (.+)/|-> Fecha estreno: \1/'
    # "(10/03/2017)" se sustituye por "|-> Fecha estreno: (10/03/2017)"

# sed -r -e 's/Dirigida por (.+)$/|-> Director: \1/'
    # "Dirigida por Gabe Polsky" se sustituye por "|-> Director: Gabe
Polsky"

# sed -r -e 's/Reparto: (.+)/|-> Reparto: \1/'
    # "Reparto: Marta Etura, Elvira Minguez, Nene..." se sustituye por "|-
> Reparto: Marta Etura, Elvira Minguez, Nene..."

# sed -r -e 's/([[:digit:]]hr [[:digit:]]*min)/|-> Duración: \1/'
    # "1hr 33min" se sustituye por "|-> Duración: 1hr 33min"

# sed -r -e '/^ +.+/d'
    # Borra las líneas que comienzan por espacio (toda la sinopsis)

# sed -r -e 's/(^[^(|)].*)/Título: \1/'
    # "Red Army" se sustituye por "Título: Red Army"
```

```
# ********** EJERCICIO 3 **********

#Script que recibe un fichero y que utiliza 'grep' y/o 'sed' para lo
siguiente:
     # Listar todos los ficheros ocultos de la carpeta personal del usuario
ordenados por número de caracteres.
     # Hacer una copia del fichero recibido en la que se eliminan todas las
líneas vacías. Su nombre será "FICH.sinLineasVacias", donde FICH es el
nombre del fichero introducido. Esto se hará siempre que exista.
     # Listar todos los procesos que el usuario está ejecutando, mostrando
el PID, la hora en que se lanzó y el nombre del fichero ejecutable.

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"

# ----------------------------------
#      CONTROL DE ARGUMENTOS
# ----------------------------------

if [ $# -ne 1 ]
then
     echo "ERROR: Debe introducir ./ejercicio3.sh <nombre_fichero.txt>"
     echo -e "\n"
     exit 1
fi



# ----------------------------------
#          PROCESAMIENTO
# ----------------------------------

directorio="/home/$USER"

echo "Listado de archivos ocultos del directorio $directorio"
temp=$(mktemp) # mktemp --> crea fichero o directorio temporal

for x in $(ls -a $directorio)
# ls -a  --> muestra todos los ficheros, incluso los ocultos
do
     nomFile=$(basename $x)
     fichOculto=$(echo $nomFile | grep '^\.')
     #Los ficheros ocultos comienzan por punto (.)

     if [[ $fichOculto == $nomFile ]]
     then
          nLineas=$(echo $fichOculto | wc -c)
          echo "$nLineas;$fichOculto" >> $temp
     fi
```


```
done
cat $temp | sort -n -k1 | sed -n -e 's/\(.*\);\(.*\)/\2/p'
#'sort' ordena de menor a mayor (-n) según la columna 1 (nLineas).
echo -e "\n"



echo "===="
echo -e "\n"
if [ -e $1 ]
then
     if [ -f "$1" ]
     then
          echo "El fichero a procesar es $1"
          cat $1 | grep '^.' > $1.sinLineasVacias
          #También podríamos usar sed -e '/^$/d'
          echo "El fichero sin líneas vacías se ha guardado en
$1.sinLineasVacias"

     else
          echo "$1 no es un fichero."
     fi

else
     echo "El fichero recogido no existe, por lo que no se copiará."
fi
echo -e "\n"



echo "===="
echo -e "\n"
echo "Listado de los procesos ejecutados por el usuario $USER: "
ps -u $USER -o pid,comm | grep '[0-9]\{1,\}' | sed -n -e 's/\([0-9]\{1,\}\)
\(.*\)/PID: \"\1\" Ejecutable: \"\2\"/p'

echo -e "\n"
```

```
# ********** EJERCICIO 4 **********

#Script que recibe dos argumentos:
     # Primer argumento: fichero de texto con direcciones IP
     # Segundo argumento: número de segundos que deben pasar antes de dar
por muerto el servidor
#Hace 'ping' a cada una de las direcciones (mandando un solo 'ping') y las
imprime en orden según lo que tardan en contestar. Si la máquina a la que
manda el 'ping' no está activa, el comando 'ping' devolverá un código de
error distinto de cero al sistema operativo.

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"

# ----------------------------------
#      CONTROL DE ARGUMENTOS
# ----------------------------------

if [ $# -ne 2 ]
then
     echo "ERROR: Debe introducir ./ejercicio4.sh <nombre_file_IP>
<segundos>"
     echo -e "\n"
     exit 1
fi

if [ ! -f $1 ]
then
     echo "$1 no es un fichero"
     echo -e "\n"
     exit 1
fi

if [ $2 -lt 1 ]
then
     echo "El número de segundos debe ser mayor o igual que 1"
     echo -e "\n"
     exit 1
fi


# ----------------------------------
#          PROCESAMIENTO
# ----------------------------------

leido=$(mktemp)
respuesta=$(mktemp)
noRespuesta=$(mktemp)
```

```
for x in $(cat $1)
do
     ping -W $2 -c 1 $x > $leido
     #El resultado de ping lo guardamos en el fichero 'leído' porque nos
hará falta en caso de que tenga éxito
     # ping:
     #     -W $2  -->  permite indicar el número de segundos ($2) a esperar
para recibir respuesta
     #     -c 1  -->  número de bloques que se envían. Ponemos 1 porque
para mandar un 'ping' debemos mandar un único bloque a cada IP

     if [ $? -eq 0 ] #Si 'ping' ha tenido éxito
     then
          cat $leido | sed -n -r -e 's/(.+from )(.+)(:.+ )(time=)(.+)(
ms)/La IP \2 ha tardado \5 milisegundos/p' >> $respuesta
          #'sed' obtiene el tiempo que ha tardado en responder y la IP, y
lo guardamos en el fichero 'respuesta'

     else #Si 'ping' no ha tenido éxito
          echo "La IP $x no respondió tras $2 segundos" >> $noRespuesta
          #En este caso no es necesario hacer 'sed' porque tenemos la IP
en $x y los segundos en $2, lo cual guardamos en el fichero 'noRespuesta'
     fi
done
sort -n -k5 $respuesta
# 'sort' ordena el fichero 'respuesta' de menor a mayor (-n) según la
columna 5 (milisegundos)

#Imprimimos las IP que no han respondido a tiempo
cat $noRespuesta

#Borramos los ficheros temporales
rm $leido
rm $respuesta
rm $noRespuesta

echo -e "\n"
```

```
# ********** EJERCICIO 5 **********

#Script que lee información de la carpeta /proc/ e imprime:
     # Modelo de procesador y Megahercios.
     # Tamaño de caché e identificador de vendedor.
     # Número máximo de hilos de ejecución.
     # Puntos de montaje activos, incluyendo el punto de montaje, el
dispositivo y el tipo de dispositivo. Ordenarlos de forma alfabética
inversa por punto de montaje.
     # Listado de todas las particiones disponibles, junto con su número de
bloques. Ordenarlas de forma alfabética inversa por partición.
#Los ficheros que se consultarán son /proc/cpuinfo, /proc/mounts y
/proc/partitions.

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"


# ----------------------------------
#   TRATAMIENTO DE /proc/cpuinfo
# ----------------------------------

#Modelo de procesador
cat /proc/cpuinfo | sed -n -r -e 's/model name[[:blank:]]*: (.+)/Modelo de
procesador: \1/p' | head -1

#Megahercios
cat /proc/cpuinfo | sed -n -r -e 's/cpu MHz[[:blank:]]*: (.+)/Megahercios:
\1/p' | head -1

#Hilos de ejecución
echo "Número de hilos de ejecución: $(cat /proc/cpuinfo | grep -E
'processor[[:blank:]]*: .+' | wc -l)"

#Tamaño de caché
cat /proc/cpuinfo | sed -n -r -e 's/cache size[[:blank:]]*: (.+)/Tamaño de
caché: \1/p' | head -1

#Identificador del vendedor
cat /proc/cpuinfo | sed -n -r -e 's/vendor_id[[:blank:]]*: (.+)/ID
vendedor: \1/p' | head -1



# ----------------------------------
#   TRATAMIENTO DE /proc/mounts
# ----------------------------------

echo -e "\n"
```

```
echo "Puntos de montaje:"
cat /proc/mounts | sed -r -e 's/([^ ]+) ([^ ]+) ([^ ]+) ([^ ]+) ([^ ]+) ([^
]+)/->Punto de montaje: \2, Dispositivo: \1, Tipo de dispositivo: \3/' |
sort -r -k2
# -r  --> inverso



# ---------------------------------
#  TRATAMIENTO DE /proc/partitions
# ---------------------------------

echo -e "\n"
echo "Particiones y número de bloques:"
cat /proc/partitions | sed -e '1,2d' | sed -n -r -e 's/ *([[:digit:]]+)
*([[:digit:]]+) *([[:digit:]]+) *([[:alnum:]]+)/->Partición: \4, Número de
bloques: \3/p' | sort -r -k2


echo -e "\n"
```

```
# ********** EJERCICIO 6 **********

#Script que muestra el contenido del fichero /etc/passwd (y parte del
fichero /etc/group) de forma amigable. Recibirá una cadena de texto. Solo se
mostrarán los usuarios que tengan esa cadena como shell de sistema
(comprobar que la cadena está justo en ese campo y no en otro). Por cada
usuario se imprimirá:
      # 'logname' del usuario
      # 'UID' del usuario
      # 'groupname' de su grupo primario
      # 'GID' de su grupo primario
      # Información 'gecos'
      # Carpeta personal
      # 'Shell' por defecto
      # Un 1 si el usuario está actualmente logueado y un 0 en caso
contrario.

#Autora: Irene Casares Rodríguez



#!/bin/bash
echo -e "\n"

# ----------------------------------
#      CONTROL DE ARGUMENTOS
# ----------------------------------

if [ $# -ne 1 ]
then
      echo "ERROR:Debe ser ./ejercicio6.sh <cadena> (por ejemplo,
/bin/bash)"
      echo -e "\n"
      exit 1
fi


# ----------------------------------
#          PROCESAMIENTO
# ----------------------------------

lineaGrupo=$[4] #La primera línea en la que aparece el campo grupo. Se irá
incrementando de 9 en 9
lineaLog=$[9] #La primera línea que indica si el usuario está logeado. Se
irá incrementando de 9 en 9



grep -E "$1$" /etc/passwd | sed -r -e
"s/^(.*):.*:(.*):(.*):(.*):(.*):(.*)$/==========\n->Logname:\1\n->UID:\2\n-
>Grupo:\n->GID:\3\n->gecos:\4\n->Home:\5\n->Shell por defecto:\6\n-
>Logueado:/" > fichero.txt
```

```
# grep -E "$1$" /ect/passwd  -->  escoge las líneas que terminen igual que
la cadena pasada, de esta forma escogemos los usuarios cuya shell de sistema
coincida con la cadena pues la shell se indica al final de la línea.
# sed -r -e ...  -->  reorganiza la información de las líneas (usuarios)
pasadas por 'grep' de una forma más "amigable"


grupoID=$(grep -E "$1$" /etc/passwd | sed -r -e
's/^.*:.*:.*:(.*):.*:.*:.*$/\1/')
usuario=$(grep -E "$1$" /etc/passwd | sed -r -e
's/^(.*):.*:.*:.*:.*:.*:.*$/\1/') #Para comprobar si está logueado



for x in $grupoID
do
      nombreGrupo=$(sed -n -r -e "s/^(.*):.*:$x:.*$/\1/p" /etc/group)
      sed -i -r -e ""$lineaGrupo"s/^(.*)$/\1$nombreGrupo/" fichero.txt

      lineaGrupo=$[$lineaGrupo+9]
done



for x in $usuario
do
      if [ "$(who | grep "^$x.*$")" ]
      then
            sed -i -r -e ""$lineaLog"s/^(.*)$/\11/" fichero.txt
      else
            sed -i -r -e ""$lineaLog"s/^(.*)$/\10/" fichero.txt
      fi

      lineaLog=$[$lineaLog+9]
done



cat fichero.txt
rm fichero.txt



echo -e "\n"
```