Programación y Administración de Sistemas
Práctica 1. Programación de *shell scripts*

Convocatoria de junio (curso 2022/2023)

Mª Isabel Jiménez Velasco

20 de febrero de 2023

Resumen

Esta serie de ejercicios se os entregan para que podáis practicar y profundicéis vuestros conocimientos de *bash* de cara al examen de prácticas. Estos ejercicios no se entregan, la evaluación de la práctica 1 se realizará mediante ejercicios similares a los expuestos en este guion. Para evitar problemas al ejecutar tus ejercicios de cara al examen, asegúrate de que todos los scripts que realices funcionen correctamente en los ordenadores de la UCO o conectándote por `ssh` al `ts.uco.es`. Para cualquier duda de los ejercicios, por favor, escribid en el foro del moodle o enviad un correo a la dirección i72jivem@uco.es

1. `ejercicio1.sh`

Desarrollar un *script* que permita generar un directorio con ficheros y subdirectorios de ejemplo que podrás utilizar en futuros ejercicios de esta práctica. El *script* recibirá 4 argumentos:

1. Ruta del nuevo directorio que se va a crear.
2. Número de subdirectorios que se crearán dentro del directorio principal.
3. Longitud mínima de los nombres de los ficheros (sin extensión) y subdirectorios.
4. Longitud máxima de los nombres de los ficheros (sin extensión) y subdirectorios.

Al ejecutarlo, deberá crear un directorio principal en la ruta que se haya especificado en el primer parámetro. Dentro de ese directorio, se deberán crear N subdirectorios (indicado por el segundo parámetro) con nombres aleatorios y de longitud aleatoria entre el mínimo y el máximo especificados como argumentos. Por último, dentro de cada uno de estos subdirectorios, se crearán 4 ficheros (vacíos) con nombres aleatorios y las extensiones .sh, .html, .key y .txt.

Por ejemplo, si el número de subdirectorios es 2 y la longitud del nombre debe estar entre 4 y 6:

```
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio1.sh carpetadeejemplo 2 4 6
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio1.sh carpetadeejemplo 2 4 6
El directorio carpetadeejemplo ya existe. Deseas eliminarlo? [y/N]
Tiempo de respuesta agotado. No se borra el directorio
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio1.sh carpetadeejemplo 2 4 6
El directorio carpetadeejemplo ya existe. Deseas eliminarlo? [y/N] N
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio1.sh carpetadeejemplo 2 4 6
El directorio carpetadeejemplo ya existe. Deseas eliminarlo? [y/N] y
Directorio carpetadeejemplo borrado con exito
```

Siendo el siguiente un ejemplo de posible directorio :

- **carpetadeejemplo**
  - fND607

    - 2Y4YF.key
    - BCdJ.sh
    - QXOAj.txt
    - zJQsdC.html
- Rreu
    - 2oHgXE.key
    - ZqCOIW.sh
    - UTBuB.txt
    - mXxp.html

En el caso de que se indique el nombre de una carpeta que ya existe, se deberá pedir confirmación para eliminarla antes de crear la nueva carpeta. Si tras **5 segundos** el usuario no indica confirmación, el script no eliminará la carpeta y pasará a la siguiente.

Para generar nombres aleatorios, puedes hacerlo de la siguiente forma:

```
tr -dc A-Za-z0-9 < /dev/urandom | head -c LONGITUD
```

Esta línea accede al fichero `/dev/urandom`, que proporciona caracteres aleatorios de manera infinita, y elimina todos aquellos que no coincidan con A-Za-z0-9 (para obtener solo cadenas alfanuméricas). Por último, con `head`, indicamos que sólo se obtenga el número de caracteres que deseamos. Ya que es posible que debas utilizar esta línea varias veces en tu *script* , sería recomendable que hagas una función que te devuelva una cadena aleatoria de los caracteres que le indiques como argumento. Recuerda realizar los controles de errores oportunos (argumentos de entrada, ...).

Nota: en Moodle tendrás disponible la carpeta de ejemplo que se ha utilizado en los siguientes ejercicios por si quieres utilizar la misma para comparar salidas.

2. `ejercicio2.sh`

Desarrolla un *script* que permita configurar los permisos de los ficheros y subdirectorios de una determinada carpeta de la siguiente forma:

- El directorio y todos los subdirectorios deberán tener todos los permisos para el usuario, lectura y ejecución para el grupo y ninguno para otros.
- Los archivos cuya extensión sea `.sh` deberán recibir permisos de ejecución para el usuario.
- Los ficheros con extensión `.key` deberán asegurarse, restringiendo los permisos de manera que sólo el usuario propietario pueda acceder a ellos.

Además, al finalizar debe mostrar una lista ordenada alfabéticamente de los usuarios que hay logeados en el sistema en el instante en el que se ejecuta el script, eliminando los usuarios repetidos (Consulta los comandos **who** y **uniq**).

A continuación, se muestra un ejemplo de ejecución sobre una carpeta generada con el *script* del ejercicio 1:

```
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio2.sh

Argumentos incorrectos. Uso: ./ejercicio2.sh <ruta_directorio>

i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio2.sh carpetadeejemplo/

Cambiando permisos de directorios...

carpetadeejemplo/
carpetadeejemplo/fND607
carpetadeejemplo/Rreu

```

```
13
14 A adiendo permisos de ejecuci n a scripts...
15
16 carpetadeejemplo/fND607/BCdJ.sh
17 carpetadeejemplo/Rreu/ZqCOIW.sh
18
19
20 Restringiendo permisos de ficheros de claves...
21
22 carpetadeejemplo/fND607/2Y4YF.key
23 carpetadeejemplo/Rreu/2oHgXE.key
24
25 Los usuarios logueados en este momento son:
26 i72jivem
27 inlmajim
```

3. `ejercicio3.sh`

Desarrolla un *script* que permita realizar una copia de seguridad de un determinado directorio y almacenarla en un fichero comprimido. El programa deberá recibir dos argumentos:

1. Directorio que se va a copiar.
2. Directorio donde se almacenará la copia comprimida.

El nombre del fichero de copia resultante deberá seguir el formato:
nombredirectorioriginal_usuario_fecha.tar.gz, donde `usuario`, es el nombre del usuario que está realizando la copia y fecha es la fecha en segundos desde el 1 de enero de 1970.

Por ejemplo, si el usuario `i72jivem` hace una copia del directorio `carpetadeejemplo` el día 20 de febrero de 2022, el fichero resultante se llamará `carpetadeejemplo_i72jivem_1676910481.tar.gz`. Para comprimir el fichero, deberás utilizar la herramienta `tar`. Consulta los argumentos necesarios para comprimir un directorio.

Si el directorio de destino de la copia no existe, deberás crearlo.

Al invocar al script, todos los ficheros de la carpeta donde se guarden los tar con una **antiguedad mayor a 200 segundos**, deberán ser borrados Además, deberás realizar los controles de errores que estimes oportunos.

Un ejemplo de ejecución es el siguiente:

```
1  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio3.sh
2  Argumentos incorrectos. Uso: ./ejercicio3.sh <directorio_origen> <directorio_destino>
3
4  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio3.sh carpetadeejemplo/ carpetaSalida
5  Copia realizada en carpetaSalida/carpetadeejemplo_i72jivem_1676910925.tar.gz
6
7  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio3.sh carpetadeejemplo/ carpetaSalida
8  Copia realizada en carpetaSalida/carpetadeejemplo_i72jivem_1676911030.tar.gz
9
10 Borrando  carpetaSalida/carpetadeejemplo_i72jivem_1676910925.tar.gz de  214  segundos
```

4. `ejercicio4.sh`

Desarrolla un *script* que permita listar todos los ficheros de un directorio con una extensión concreta sin mostrar los subdirectorios pero incluyendo los ficheros que estos puedan contener.

El script recibirá dos argumentos: el directorio y la extensión. El nombre del fichero deberá mostrarse sin su ruta, solo incluyendo el nombre. Además, se deberá añadir un número que indicará el orden de cada fichero y también otro número que indicará el número de caracteres del mismo así como el número de veces que aparece un determinado caracter que se le solicita al usuario. Si el usuario no indica el caracter en 5 segundos, utiliza la letra "a".

A continuación se muestra un ejemplo de ejecución:

```
1
2  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio4.sh carpeta/ html
3  carpeta/ no es un directorio.
4
5  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio4.sh carpetadeejemplo/
6  Argumentos incorrectos. Uso: ./ejercicio4.sh <ruta_directorio><extension_fichero>
7
8  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio4.sh carpetadeejemplo/ html
9  Que caracter quieres contar?
10 Lo introducido no es un caracter, o no se ha intoducido luego se quedara caracter por defecto :  a
11
12  Ficheros:
13
14      1            mXxp.html         9                 0
15      2            zJQsdC.html       11                0
16
17 i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio4.sh carpetadeejemplo/ html
18 Que caracter quieres contar? s
19 Caracter recibido  s
20
21  Ficheros:
22
23      1            mXxp.html         9                 0
24      2            zJQsdC.html       11                1
```

Nota: la herramienta `nl` puede servirte de utilidad para numerar las líneas.

5. `ejercicio5.sh`

Crear un script que reciba como argumento un parámetro que será un directorio y un segundo argumento que será un número (número en bytes). El script debebuscar todos los ficheros que estén alojados en la carpeta que se pasa como primer argumento cuyo tamaño total en bytes sea mayor o igual que el número pasado como segundo argumento. Para cada fichero deberá mostrar:

1. El nombre del fichero sin la ruta
2. La fechar de creación (legible)
3. Tamaño en bytes
4. La cadena de permisos del fichero
5. Un 1 si el fichero es ejecutable y un 0 sino lo es

Una ejecución del *script* debe producir una salida similar a (No olvide mostrar los ficheros de **mayor a menor tamaño en bytes**):

```
1  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio5.sh
2  Argumentos incorrectos. Uso: ./ejercicio5.sh <ruta_directorio> <tam_bytes>
3
4  i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio5.sh carpeta/ 88
5  carpeta/ no es un directorio.
6
7  i72jivem@VTS1:~/Desktop/PAS_practicas/p1$ ./ejercicio5.sh carpetadeejemplo/ 0
8  El n mero de bytes debe ser positivo.
9
10 i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio5.sh ejemploCarpeta/ 35
11 Nombre, FeachaModificacion, Tamano,  Permisos, Ejecutale
12 libglib-2.0.so.0; 2016-02-28 19:08:56.000000000 +0100; 822344; -rw-r--r--; 0
13 elinfiernoexiste.jpg; 2016-02-28 19:09:02.000000000 +0100; 119847; -rw-r--r--; 0
14 raid0.png; 2016-02-28 19:09:18.000000000 +0100; 50683; -rw-r--r--; 0
15 logo_eps_grande.jpg; 2016-02-28 19:09:10.000000000 +0100; 43161; -rw-r--r--; 0
16 libsysfs.so.2; 2016-02-28 19:09:30.000000000 +0100; 38644; -rw-r--r--; 0
17 HD.png; 2016-02-28 19:09:14.000000000 +0100; 36710; -rw-r--r--; 0
18 disco2.png; 2016-02-28 19:09:14.000000000 +0100; 36710; -rw-r--r--; 0
19 devocion-administrador-de-sistemas.png; 2016-02-28 19:09:08.000000000 +0100; 21613; -rw-r--r--; 0
20 Logo_uco.gif; 2016-02-28 19:09:10.000000000 +0100; 20005; -rw-r--r--; 0
21 audit.h; 2017-02-24 12:46:02.000000000 +0100; 18221; -rw-r--r--; 0
22 auditBackup.h; 2017-02-24 12:46:54.000000000 +0100; 18221; -rw-r--r--; 0
```

```
Linus_torvalds.jpg; 2016-02-28 19:09:04.000000000 +0100; 17670; -rw-r--r--; 0
libhandle.so.1.0.3; 2016-02-28 19:09:26.000000000 +0100; 10760; -rw-r--r--; 0
centos.png; 2016-02-28 19:09:16.000000000 +0100; 8215; -rw-r--r--; 0
eje2; 2016-02-28 19:09:34.000000000 +0100; 7275; -rwxr--r--; 1
a.out; 2016-02-28 19:08:56.000000000 +0100; 7275; -rwxr--r--; 1
a2.out; 2016-02-28 19:09:14.000000000 +0100; 7275; -rwxr--r--; 1
debian.png; 2016-02-28 19:09:16.000000000 +0100; 2626; -rw-r--r--; 0
fedora.png; 2016-02-28 19:09:18.000000000 +0100; 2386; -rw-r--r--; 0
ecoc.h; 2016-02-28 19:09:24.000000000 +0100; 2377; -rw-r--r--; 0

....

i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio5.sh ejemploCarpeta/ 2700
Nombre, FeachaModificacion, Tamano,  Permisos, Ejecutale
libglib-2.0.so.0; 2016-02-28 19:08:56.000000000 +0100; 822344; -rw-r--r--; 0
elinfiernoexiste.jpg; 2016-02-28 19:09:02.000000000 +0100; 119847; -rw-r--r--; 0
raid0.png; 2016-02-28 19:09:18.000000000 +0100; 50683; -rw-r--r--; 0
logo_eps_grande.jpg; 2016-02-28 19:09:10.000000000 +0100; 43161; -rw-r--r--; 0
libsysfs.so.2; 2016-02-28 19:09:30.000000000 +0100; 38644; -rw-r--r--; 0
HD.png; 2016-02-28 19:09:14.000000000 +0100; 36710; -rw-r--r--; 0
disco2.png; 2016-02-28 19:09:14.000000000 +0100; 36710; -rw-r--r--; 0
devocion-administrador-de-sistemas.png; 2016-02-28 19:09:08.000000000 +0100; 21613; -rw-r--r--; 0
Logo_uco.gif; 2016-02-28 19:09:10.000000000 +0100; 20005; -rw-r--r--; 0
audit.h; 2017-02-24 12:46:02.000000000 +0100; 18221; -rw-r--r--; 0
auditBackup.h; 2017-02-24 12:46:54.000000000 +0100; 18221; -rw-r--r--; 0
Linus_torvalds.jpg; 2016-02-28 19:09:04.000000000 +0100; 17670; -rw-r--r--; 0
libhandle.so.1.0.3; 2016-02-28 19:09:26.000000000 +0100; 10760; -rw-r--r--; 0
centos.png; 2016-02-28 19:09:16.000000000 +0100; 8215; -rw-r--r--; 0
eje2; 2016-02-28 19:09:34.000000000 +0100; 7275; -rwxr--r--; 1
a.out; 2016-02-28 19:08:56.000000000 +0100; 7275; -rwxr--r--; 1
a2.out; 2016-02-28 19:09:14.000000000 +0100; 7275; -rwxr--r--; 1
```

6. `ejercicio6.sh`

Desarrolla un *script* que simule la creación de nuevos usuarios. La gestión de usuarios es una tarea muy común para un administrador de sistemas. Sin embargo, en los servidores de la universidad no tenemos la posibilidad de crear o eliminar usuarios. Por ello, en este ejercicio se pretende crear un sistema de usuarios, muy sencillo, que simule el sistema utilizado en Linux.

El sistema constará de un fichero (por ejemplo `users.txt`) que almacenará los usuarios existentes. Por otro lado, cada usuario tendrá su home dentro de un directorio (por ejemplo dentro de `./homes`). Además, habrá un directorio que contendrá los ficheros por defecto que se añaden al home de un usuario al crearlo (por ejemplo `./skel`).

Dentro de nuestro *script* deberemos implementar una función `crear_usuario` que se encargue de añadir un nuevo usuario al sistema con el nombre indicado en su primer argumento. Al crearlo, lo añadirá al fichero de texto, le creará su home y meterá los archivos por defecto que se encuentren en el directorio `skel`. Si se intenta crear un usuario que ya existe, no deberá volver a crearlo.

Una vez creada dicha función, el script deberá llamarla utilizando como nombre el primer argumento con el que se invoque el *script* . Recuerda realizar los controles de errores oportunos.

A continuación se muestra un ejemplo de ejecución del *script* :

```
i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio6.sh
Argumentos incorrectos. Uso: ./ejercicio6.sh <usuario>

i72jivem@VTS1:~/Desktop/PAS/p1$ mkdir skel
i72jivem@VTS1:~/Desktop/PAS/p1$ touch skel/file1
i72jivem@VTS1:~/Desktop/PAS/p1$ touch skel/file2
i72jivem@VTS1:~/Desktop/PAS/p1$ touch skel/file3

i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio6.sh user1
Se ha creado el usuario user1.

i72jivem@VTS1:~/Desktop/PAS/p1$ cat users.txt
user1
i72jivem@VTS1:~/Desktop/PAS/p1$ ls homes/user1/
file1  file2  file3

i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio6.sh user2
Se ha creado el usuario user2.
```

```
19
20 i72jivem@VTS1:~/Desktop/PAS/p1$ cat users.txt
21 user1
22 user2
23 i72jivem@VTS1:~/Desktop/PAS/p1$ ls homes/user2/
24 file1  file2  file3
25
26 i72jivem@VTS1:~/Desktop/PAS/p1$ ./ejercicio6.sh user1
27 El usuario user1 ya existe.
```

A. Conexión en remoto a la UCO

Para poder trabajar en remoto en el servidor de la UCO, necesitaremos, por un lado, conectarnos a una sesión de `ssh` para poder tener una terminal remota y, por otro lado, conectarnos por `sftp` para poder transferir archivos.

A.1. Conexión SSH

A.1.1. Linux

Para conectarnos por `ssh` desde Linux, basta con abrir una terminal y escribir:

```
1 ssh usuarioUCO@ts.uco.es
```

En caso de que no tengamos el cliente de `ssh` instalado, deberemos instalarlo con

```
1 apt install openssh-client
```

A.1.2. Windows 10+

En primer lugar es necesario habilitar el cliente `ssh`, que viene deshabilitado por defecto. Para ello, puedes seguir estas instrucciones [1]. Una vez habilitado, basta con abrir una ventana de `cmd` y escribir el comando:

```
1 ssh usuarioUCO@ts.uco.es
```

A.2. Conexión SFTP

A.2.1. Linux

Para conectarnos por `sftp` desde Linux, basta con abrir una ventana del explorador de archivos y en la barra de direcciones escribir:

```
1 sftp://usuarioUCO@ts.uco.es/home/usuarioUCO
```

A.2.2. Windows 10+

En el caso de Windows deberemos instalar algún cliente de `sftp` como por ejemplo WinSCP. El usuario de conexión será nuestro usuario de la UCO y la dirección del servidor `ts.uco.es`.

[1] `https://www.howtogeek.com/336775/how-to-enable-and-use-windows-10s-built-in-ssh-commands/`