SESIÓN TP2

Sistemas Operativos 1
Febrero 2025

1 INTRODUCCIÓN

En esta sesión teórico-práctica se presentan dos comandos (`wc` y `awk`) relacionados con el primer tutorial y primera práctica de la asignatura. A continuación, se proponen algunos ejercicios para comprender su utilidad.

2 REVISIÓN DE LOS SCRIPTS PRESENTADOS EN LA TP1

Revisamos primero algunas de las soluciones enviadas por vuestros compañer@s en la TP1.

3 REDIRECCIONAMIENTO DE LA ENTRADA-SALIDA

La línea de comandas nos ofrece un gran número de aplicaciones "pequeñas". En lugar de tener una aplicación "grande" que puede realizar muchas tareas diferentes, se disponen de aplicaciones pequeñas especializadas en una determinada tarea. Se puede crear una funcionalidad compleja encadenando las funcionalidades simples de las aplicaciones, una detrás de otra, utilizando el redireccionamiento de entrada-salida.

En la primera práctica de laboratorio nos centramos en el **redireccionamiento hacia un archivo o desde un archivo**. En la segunda práctica veremos la **tubería (pipe o canonada)** también muy útil para encadenar aplicaciones entre sí. En algunas aplicaciones con interfaz gráfica, se realiza la comunicación con otras aplicaciones que hay por debajo utilizando estas herramientas de redireccionamiento.

En esta sección empezaremos a ver el potencial que nos ofrece el redireccionamiento y que recibirá más atención a la segunda práctica.

wc

Cogemos como ejemplo la aplicación `wc` (word count), una aplicación que permite obtener de un archivo de texto el número de líneas, palabras y caracteres que tiene. Veremos esta aplicación con más detalle en la práctica 2. Por el momento haremos algunas pruebas con esta aplicación.

Abrir un terminal y ejecute la aplicación `wc`, seguido de un texto cualquiera. Para terminar, pulse Ctrl+D.

```
$wc
Prueba de la aplicación
Word count
Sesión de problemas
```

Esto hará que se imprima por pantalla información sobre el texto introducido

```
3 8 52
```

Hay 3 líneas, 8 palabras y 52 caracteres en el texto introducido. Intente hacer un archivo de texto plano que contenga un texto cualquiera. Entonces ejecute en la línea de pedidos la siguiente instrucción

```
$ wc < archivo.txt
```

¿Qué hace esta instrucción? A continuación, ejecute la siguiente instrucción

```
$ wc < archivo.txt > resultado.txt
```

¿Qué es lo que hace la última instrucción?

awk

`awk` es una herramienta excelente para crear scripts de shell de UNIX/Linux. Se trata de una aplicación con su propio lenguaje de programación diseñado para procesar datos de texto plano. La aplicación se puede utilizar, por ejemplo, para procesar las columnas de un texto. A continuación se muestran algunos ejemplos que muestran la funcionalidad por la que se utilizará esta aplicación en la siguiente práctica: la extracción de una terminada columna de un archivo de texto.

Podemos extraer una de las columnas que se han guardado en el archivo `resultado.txt` en el experimento anterior. Por ejemplo, para obtener el número de caracteres (que se encuentra en la 3ª columna) haremos:

```
$ awk '{print $3}' resultado.txt
```

El valor obtenido puede asignarse a una variable

```
$ valor=$(awk '{print $3}' resultado.txt)
```

A continuación se muestra otro ejemplo, script `script_awk.sh`

```
#!/bin/bash
if [ $# -ne 1 ]
then
     echo "Número incorrecto de parámetros"
     exit 1
fin

archivo=$1
col1=($(awk '{print $1}' $fichero)) // punto 1
col2=($(awk '{print $2}' $archivo))
len=${#col1[*]} // punto 2

e=0
sum1=0
sum2=0

while [ $i -lt $len ]
don
   sum1=$(($sum1+${col1[$i]})) // punto 3
   sum2=$(($sum2+${col2[$i]}))
   (( i++ ))
done
echo $sum1 $sum2
exit 0
```

¿Qué es lo que hace el código anterior? En particular, ¿qué es lo que se hace en el punto 1, 2 y 3 del código?

Realiza los siguientes experimentos:

1. Ejecuta el código utilizando como parámetro el archivo de texto `awk-text.txt`.
2. Modifica el script para calcular cuántos números de la primera columna son superiores al de la segunda.

Vamos a utilizar este script como base para calcular el tamaño, en bytes, que ocupan los archivos (y subdirectorios) de un determinado directorio. No hace falta implementar recursividad.

En particular, el script tendrá sólo un parámetro, el directorio a analizar

```
$ ./script <directorio>
```

Para implementarlo se recomienda seguir los siguientes pasos:

1. El script debe hacer un `ls -l <directorio>` y redirigir el resultado a un archivo temporal donde se ejecuta el script. Supongamos que el directorio existe y que el directorio especificado como argumento es diferente al que se ejecuta el script (así evitamos problemas con el archivo temporal que se genera).
2. Utiliza `awk` para extraer del archivo temporal la columna que te interesa.
3. Calcula la suma e imprime el resultado por pantalla.
4. Elimina el archivo temporal.