P2. COMANDAS PARA MANIPULAR FICHEROS

Sistemas Operativos 1
Marzo 2025

1 INTRODUCCIÓN

Hemos visto en la práctica anterior que *bash scripting* representa una herramienta esencial para los profesionales de la informática, permitiendo automatizar tareas repetitivas y procesar datos eficientemente en sistemas Unix. Mediante *scripts bash*, es posible también manipular archivos, filtrar información y transformar datos de manera sistemática, habilidades fundamentales para cualquier ingeniero informático.

Esta práctica se centra en el procesamiento de datos relacionados con el ODS 6 "Agua limpia y saneamiento". Este objetivo busca garantizar la disponibilidad de agua, su gestión sostenible y el saneamiento para todos. El monitoreo de la calidad del agua es crucial para detectar contaminantes, evaluar tratamientos y asegurar el acceso a agua potable segura.

Los ejercicios propuestos permitirán aplicar comandos como `find`, `grep`, `sed`, `awk` y `sort`, etc. para analizar datos de calidad del agua, desarrollando habilidades técnicas mientras se trabaja con información relevante para uno de los mayores desafíos ambientales actuales.

2 SCRIPTS A PROGRAMAR

A continuación, se presentan 6 ejercicios, cada uno con la misma puntación. Todos los scripts necesitan argumentos por líneas de comando, a menos que se indique lo contrario. En caso de que no se introduzcan el número necesario de argumentos, el script deberá mostrar un mensaje de error por pantalla y salir con el código de error `exit 1`. En caso de que se especifiquen todos los argumentos, puede suponer que se han especificado de forma correcta. Si el script se ejecuta correctamente, será necesario que salga con un código de ejecución correcta, `exit 0`.

Todos los nombres de ficheros, directorios y cadenas a tratar no tendrán espacios ni caracteres especiales, y podemos suponer que todos los ficheros a tratar son ficheros de texto plano. Al campus virtual hay disponible una carpera comprimida (**pruebas**) que contiene un gran número de ficheros de texto y directorios y que se utilizarán para realizar las pruebas correspondientes.

Para realizar estos ejercicios **está permitido utilizar las instrucciones que hay en el tutorial 1 y tutorial 2** (a menos que se indique lo contrario), así como utilizar opciones de estas instrucciones, aunque no estén incluidas en los tutoriales. Se recomienda investigar las opciones mediante el comando `man`. El uso de comandos no permitidos tendrá una penalización.

Ejercicio 1

Escribe un script llamado **ejercicio1.sh** que tome como argumento un archivo de datos de calidad de agua y muestre el número total de muestras, cuántas muestras están por debajo del nivel mínimo permitido de pH (pH < 6.5) y cuántas están por encima del nivel máximo permitido (pH > 8.5). El script debe validar que el archivo existe y tiene el formato correcto. Se prové del archivo `calidad_agua.csv` para la realización de pruebas.

Formato de uso: `./ejercicio1.sh <archivo_calidad_agua>`

Ejemplos de uso:

```
./ejercicio1.sh
Se requiere el siguiente formato: ./ejercicio1.sh <archivo_calidad_agua>

./ejercicio1.sh archivo_inexistente.csv
El archivo 'archivo_inexistente.csv' no existe.

./ejercicio1.sh archivo_invalido.txt
El archivo no tiene el formato correcto. Debe contener una columna de
pH.

./ejercicio1.sh calidad_agua.csv
Total de muestras analizadas: 10
Muestras con pH por debajo de 6.5: 3
Muestras con pH por encima de 8.5: 3
```

Ejercicio 2

Crea un script llamado **ejercicio2.sh** que procese un archivo de registro de consumo de agua diario de diferentes hogares y genere un informe con el consumo promedio por hogar, identificando aquellos que superan un umbral definido como argumento (en litros). Si no se proporciona umbral, usa 200 litros como valor predeterminado. El script debe poder filtrar por mes si se proporciona este argumento opcional. Se prové del archivo `consumo_agua.txt` para realizar pruebas.

Formato de uso: `./ejercicio2.sh <archivo_consumo> [umbral] [mes]`

Ejemplos de uso:

```
./ejercicio2.sh
Uso: ./ejercicio2.sh <archivo_consumo> [umbral] [mes]
  - archivo_consumo: archivo con datos de consumo de agua
  - umbral: consumo máximo en litros (opcional, default: 200)
  - mes: filtrar por mes en formato 'MM' (opcional)

./ejercicio2.sh consumo_agua.txt
Consumo promedio de agua por hogar:
--------------------------------
H001: 177.50 litros
H002: 231.67 litros
H003: 147.50 litros

Hogares que superan el umbral de 200 litros:
--------------------------------
H002: 231.67 litros
```

```
Período analizado: 2023-01-01 a 2023-02-03

./ejercicio2.sh consumo_agua.txt 180 01
Consumo promedio de agua por hogar:
--------------------------------
H001: 181.67 litros
H002: 236.67 litros
H003: 151.67 litros

Hogares que superan el umbral de 180 litros:
--------------------------------
H001: 181.67 litros
H002: 236.67 litros

Período analizado: Mes 01 de 2023
```

Ejercicio 3

Desarrolla un script llamado **ejercicio3.sh** que analice un directorio con múltiples archivos de datos de precipitaciones (uno por estación meteorológica) y genere un informe con las estaciones que registran sequía (precipitación mensual inferior a 30mm) y aquellas con riesgo de inundación (precipitación diaria superior a 50mm). El script debe recibir como argumento la ruta del directorio con los archivos de datos. Se prové de un directorio `datos_precipitaciones` con algunos archivos para realizar pruebas.

Formato de uso: `./ejercicio3.sh <directorio_datos>`

Ejemplos de uso:

```
./ejercicio3.sh
Uso: ./ejercicio3.sh <directorio_datos>

./ejercicio3.sh directorio_no_existente
Error: El directorio 'directorio_no_existente' no existe.

./ejercicio3.sh datos_precipitaciones
Analizando datos de precipitaciones...
================================================
ESTACIONES CON RIESGO DE SEQUÍA (precipitación total < 30mm):
------------------------------------------------
estacion_sur: 11.2 mm

ESTACIONES CON RIESGO DE INUNDACIÓN (precipitación diaria > 50mm):
------------------------------------------------
estacion_este:
2023-01-04: 55.2 mm
estacion_norte:
2023-01-05: 62.7 mm

ESTADÍSTICAS GENERALES:
------------------------------------------------
Total de estaciones analizadas: 3
Estaciones con riesgo de sequía: 1
Estaciones con riesgo de inundación: 2
```

Ejercicio 4

Implementa un script llamado **ejercicio4.sh** que simule un gestor de procesos para una planta de tratamiento de agua. El script debe mostrar al usuario los procesos activos de tratamiento (simulados con procesos del sistema), permitiendo detener, reiniciar o cambiar la prioridad de los mismos. Deberá mostrar un menú interactivo con estas opciones y validar las entradas del usuario.

El script debe:

1. Mostrar los primeros 5 procesos del sistema (más la cabecera) utilizando el comando `ps`
2. Crear un menú interactivo con opciones numeradas usando estructuras de control como `case` y bucles `while`. La captura de la opción del usuario se realizará mediante `read`.
3. Verificar que las entradas del usuario sean válidas antes de procesar cualquier acción.
4. Comprobar que existe el PID introducido usando kill y verificar que los valores de prioridad estén en el rango de -20 a 19.
5. Las funcionalidades del script incluyen:
    a. Detener proceso mediante `kill`.
    b. Cambiar prioridad usando `renice`.
    c. Para simular el reinicio de un proceso basta con un mensaje informativo.
    d. Para las estadísticas se mostrará información del sistema como uso de memoria y CPU usando `free`, `uptime` , `df` o `top` como se muestra en el ejemplo de uso.
    e. Implementar captura y manejo de errores para todas las operaciones. Redirigir errores a /dev/null cuando sea apropiado. Mostrar mensajes de error informativos cuando falle una operación.

Formato de uso: `./ejercicio4.sh`

Ejemplo de uso:

```
./ejercicio4.sh
========================================
  GESTOR DE PROCESOS - PLANTA DE TRATAMIENTO DE AGUA
========================================

Procesos activos de tratamiento:
ID  PRI     TIEMPO          COMMAND
1   0       00:32:15        systemd
2   0       00:28:45        bash
3   0       00:15:23        sshd
4   0       00:10:05        logger
5   0       00:05:12        ps

Opciones:
1. Detener un proceso
2. Cambiar prioridad de un proceso
3. Reiniciar un proceso
4. Ver estadísticas de la planta
5. Salir

Seleccione una opción (1-5): 2

Introduzca el ID del proceso: 1234
Introduzca el nuevo valor de prioridad (-20 a 19, menor es más
prioritario): 10
Cambiando prioridad del proceso 1234 a 10...
```

```
Prioridad cambiada correctamente.


./ejercicio4.sh
========================================
  GESTOR DE PROCESOS - PLANTA DE TRATAMIENTO DE AGUA
========================================

Procesos activos de tratamiento:
    PID  NI     ELAPSED COMMAND
      1   0       02:45 systemd
      2   0       02:45 kthreadd
      3 -20       02:45 rcu_gp
      4 -20       02:45 rcu_par_gp
      5 -20       02:45 slub_flushwq

Opciones:
1. Detener un proceso
2. Cambiar prioridad de un proceso
3. Reiniciar un proceso
4. Ver estadísticas de la planta
5. Salir

Seleccione una opción (1-5): 4

Estadísticas de la Planta de Tratamiento de Agua
------------------------------------------------
Uso de memoria:
Mem:            6.8Gi        814Mi        5.1Gi         39Mi        880Mi
5.7Gi
Swap:             0B           0B           0B

Carga del sistema:
 19:42:05 up 2 min,  1 user,  load average: 0.52, 0.43, 0.18

Uso de disco:
/dev/mapper/ubuntu--vg-ubuntu--lv   28G   14G   14G  51% /

Procesos más intensivos en CPU:

USER           PID %CPU %MEM     VSZ    RSS TTY       STAT START    TIME
COMMAND
oslab         2038 13.5  4.5 5057576 325628 ?       Ssl  19:39    0:19
/usr/bin/gnome-shell
oslab         2142  2.8  1.1 1001448 80872 ?        Sl   19:39    0:04
/usr/bin/nautilus --gapplication-service
oslab         1951  1.9  0.4 719228 33720 ?         SNsl 19:39    0:02
/usr/libexec/tracker-miner-fs-3
root           853  1.3  0.1 236944 10808 ?         Ssl  19:39    0:02
/usr/libexec/polkitd --no-debug
root             1  1.2  0.1 166948 12400 ?         Ss   19:39    0:02
/sbin/init

Presione Enter para continuar...
```

Ejercicio 5

Crea un script llamado **ejercicio5.sh** que, dado un directorio como argumento, organice automáticamente los archivos de datos hídricos en subdirectorios según su categoría (calidad, consumo, precipitaciones, etc.) identificada en el nombre del archivo. El script debe crear los subdirectorios necesarios, mover los archivos correspondientes y generar un informe de la organización realizada. Se prové del directorio `datos_hidricos` para realizar pruebas.

Formato de uso: `./ejercicio5.sh <directorio_datos>`

Ejemplos de uso:

```
./ejercicio5.sh
Uso: ./ejercicio5.sh <directorio_datos>

./ejercicio5.sh directorio_inexistente
Error: El directorio 'directorio_inexistente' no existe.

./ejercicio5.sh
Uso: ./ejercicio5.sh <directorio_datos>

./ejercicio5.sh directorio_inexistente
Error: El directorio 'directorio_inexistente' no existe.


./ejercicio5.sh datos_hidricos
Iniciando organización de archivos de datos hídricos...
Directorio creado: datos_hidricos/calidad
Directorio creado: datos_hidricos/consumo
Directorio creado: datos_hidricos/precipitaciones
Directorio creado: datos_hidricos/niveles
Movido: calidad_rio_norte_ene2023.csv -> calidad/
Movido: calidad_rio_sur_ene2023.csv -> calidad/
Movido: consumo_ciudad_A_2023.txt -> consumo/
Movido: consumo_ciudad_B_2023.txt -> consumo/
Movido: precipitaciones_este_enero.dat -> precipitaciones/
Movido: precipitaciones_oeste_enero.dat -> precipitaciones/
Movido: niveles_acuifero_este_2023.xlsx -> niveles/
Movido: niveles_acuifero_oeste_2023.xlsx -> niveles/

Organización completada.
------------------------
Total de archivos procesados: 8
Archivos movidos: 8
Categorías detectadas: 4
Se ha generado un informe en: datos_hidricos/informe_organizacion.txt
```

Ejercicio 6

Realizar un script llamado **ejercicio6.sh** que, dado un directorio y dos cadenas de caracteres como argumentos, busque en todos los archivos de texto dentro del directorio (y sus subdirectorios) aquellos que contengan la primera cadena (parámetro de calidad del agua) y muestre las líneas donde el valor asociado a ese parámetro supere el umbral especificado por la segunda cadena. El script deberá mostrar el nombre del archivo, el nombre de la estación de monitoreo y el valor encontrado, ordenados de mayor a menor valor. Se prové del directorio `datos_agua` para la realización de pruebas.

Formato de uso: `./agua_monitoreo.sh <directorio> <parametro> <umbral>`

Ejemplos de uso:

```
./agua_monitoreo.sh datos_agua turbidez 5.0
Estaciones con turbidez superior a 5.0:
-----------------------------------------
Archivo: estacion4.txt
Estación: Canal Industrial
Valor de turbidez: 12.5
-----------------------------------------
Archivo: estacion2.txt
Estación: Embalse Central
Valor de turbidez: 9.7
-----------------------------------------
Archivo: estacion5.txt
Estación: Arroyo Este
Valor de turbidez: 5.6
-----------------------------------------

./agua_monitoreo.sh datos_agua pH 7.0
Estaciones con pH superior a 7.0:
-----------------------------------------
Archivo: estacion3.txt
Estación: Lago Municipal
Valor de pH: 8.4
-----------------------------------------
Archivo: estacion5.txt
Estación: Arroyo Este
Valor de pH: 7.5
-----------------------------------------
Archivo: estacion1.txt
Estación: Río Grande Norte
Valor de pH: 7.2
-----------------------------------------
```

3 INFORME

Además de entregar los scripts correspondientes a los ejercicios anteriores, se requiere la entrega de un informe sobre el trabajo realizado. **El informe se debe entregar en formato PDF** (NO se admiten informes en formato doc, docx, odt, etc). Este informe debe incluir una **introducción**, una sección donde se muestran y comentan las **pruebas realizadas y respuestas a las preguntas**, así como una sección con las **conclusiones y valoraciones personales de la práctica.** Importante: **se ha de indicar en una sección la contribución de cada uno de los miembros del grupo, indicando el porcentaje de la nota que merecería cada uno (p.ej., 25%-25%-50%; 50%-50%; 30%-70%).**

El documento ha de tener un **máximo de 4 páginas** (sin contar la portada o índice). Se evaluará las pruebas realizadas (60%), escritura y expresión (10%), así como la estructura y presentación del informe (30%).

4 ENTREGA Y EVALUACIÓN

Cada grupo debe subir un **único archivo ZIP** a la tarea asociada a esta práctica en el campus virtual con el nombre **P2_GrupoXX_nombre1_apellido1_nombre2_apellido2.zip**, y debe incluir los **scripts** y el **informe** solicitado con todas las secciones especificadas en el apartado anterior.

La nota final de la práctica tendrá los siguientes pesos: 60% scripts y 40% informe.