I.E.S. Fernando Aguilar Quignon
C/Conil de la Frontera, 3
CP 11010, Cádiz

Administración de Sistemas Operativos - 1ª Evaluación (RA 7 – CE a, c)

Unidad Didáctica 2. Lenguajes de script

1. Crea un script, de nombre *ejemplosArgumentos.sh*, que reciba un número cualquiera de argumentos. El script deberá mostrar por pantalla la siguiente información:

- Nombre del script.
- Número de argumentos recibidos.
- Valor de todos los argumentos recibidos.

2. Crea un script, de nombre *comparador.sh*, que reciba dos números por la entrada estándar y los compare. Tras compararlos enviará a la salida estándar un mensaje del tipo "El primer número (…) es mayor que el segundo número (…)", "El primer número (…) es menor que el segundo número (…)" o "El primer número (…) es igual que el segundo número (…)". Comprobar el funcionamiento introduciendo los números a través del teclado.

3. Crea un script, de nombre *parimpar.sh*, que solicitará un número al usuario, mediante la entrada estándar, y comprobará si el número es par o impar.

4. Crea un script, de nombre *comprobarVarSesion.sh*, que preguntará al usuario, mediante la entrada estándar, el nombre de una variable de entorno. Una vez que tenga el nombre comprobará la existencia de la misma, mostrándole un mensaje al usuario sobre el resultado.

5. Modifica el script anterior para que, además de comprobar la existencia, muestre por pantalla el valor de la variable solicitada.

6. Modifica el script anterior para que el usuario introduzca el path de un fichero. Este contendrá los nombres de variables, separadas por ':', que serán comprobadas y mostradas. Las variables podrán encontrarse en la misma línea o en diferentes.

7. Crea un script, de nombre *compartir.sh*, que sirva para copiar ficheros en un directorio compartido. Recibirá como parámetro de entrada el path de un fichero y realizará una copia en /tmp/<login usuario>. Además de lo anterior, deberá prestar especial atención a no sobreescribir ficheros ya compartidos por el usuario. El script creará el directorio si no existiera.

8. Modifica el script anterior para que:

- Antes de realizar la copia, compruebe si el fichero recibido como argumento existe realmente. En caso de que no exista mostrar el mensaje "ERROR: El fichero <path del fichero> no existe".
- Debe comprobar el valor de retorno del comando empleado para hacer la copia. En función de valor del retorno, el script enviará a la salida estándar el mensaje "Copia realizada correctamente" o "Se detecto un error durante la copia".

9. Crea un script, de nombre *contenido.sh*, que reciba un único argumento. Si el valor recibido es un nombre de fichero existente, deberá paginar su contenido. En cambio, si el valor recibido es un nombre de directorio, deberá listar su contenido.

10. Modifica el script anterior para que compruebe si ha recibido el argumento necesario. En caso de no recibir argumento, mostrará el mensaje: "USO: <nombre del script> <nombre de fichero o directorio>" y terminará su ejecución devolviendo un 1. En caso contrario, un 0.

11. Crea un script, de nombre *ejecutablesDeUsuario.sh,* que comprobará si el usuario que ejecuta el script tiene un directorio llamado "ejecutables" en su directorio casa. Solo en caso de que el usuario no disponga de este directorio, el script procederá a crearlo, y añadirá, de forma persistente, al valor de la variable PATH la ruta absoluta del nuevo directorio. De esta forma, el usuario podrá colocar en su directorio "ejecutables" los programas que quiera ejecutar desde cualquier ubicación, sin necesidad de tener que teclear su ruta absoluta o relativa.

12. Crea un script, de nombre *usuariosConectados.sh*, que reciba como parámetro un nombre de usuario y escriba por pantalla si el usuario se encuentra conectado al sistema y desde cuándo. Deberá comprobar primeramente si el usuario existe en el sistema.

13. Crea un script, de nombre *opcionesTXT.sh*, que reciba dos argumentos obligatorios. El primero consistirá en una opción de entre las siguientes: -u, -p y -l. El segundo será el nombre de un fichero regular. Según el valor recibido en el primer argumento, el script editará, paginará o mostrará el número de líneas del fichero recibido como segundo parámetro. Con respecto a las comprobaciones, el script deberá asegurarse que las opciones son adecuadas y que el fichero es un fichero y que existe.

14. Modifica el script anterior para que permita introducir las opciones del primer argumento tanto en mayúsculas como en minúsculas.

15. Modifica el anterior script para que, justo antes de editar el fichero, compruebe si el usuario tiene permiso de lectura y escritura.

16. Crea un script, de nombre *numeraLineas.*sh, que recibirá como único argumento el nombre de un fichero de texto. El script enviará a la salida estándar las líneas del fichero recibido como argumento, numerándolas de forma ascendente y consecutiva.

17. Crea un script, de nombre *pingMultiple.*sh, que reciba como único argumento el nombre de un fichero de texto. Suponiendo que el fichero incluya una dirección ip en cada línea, el script deberá comprobar si obtiene respuesta al enviarles un comando ping.

18. Crea un script, de nombre *usuariosLocales.sh*, que recorra el fichero /etc/passwd y muestre el nombre de usuario con su UID. El UID deberá ser mayor o igual a 1000.

19. Modifica el anterior script para que compruebe a qué usuarios locales del sistema se le ha creado directorio casa.

20. El comando cat concatena el contenido de los ficheros que reciba como argumento, enviando el resultado a la salida estándar. Sin embargo, no establece ninguna separación entre el contenido de cada fichero concatenado, por lo que puede resultar difícil identificar cada fragmento del resultado. En este ejercicio se pide crear un script de bash, de nombre *catConSeparador.sh*, que reciba como argumento uno o más nombres de ficheros regulares. El script realizará las siguientes comprobaciones y acciones:

- Comprobará que todos los argumentos recibidos son ficheros regulares. En caso contrario mostrará el mensaje "ERROR: <argumento> no es un nombre de fichero regular", y terminará devolviendo un 1.
- Solicitará que el usuario introduzca por teclado la cadena de caracteres que desea emplear como separador de ficheros para la concatenación.
- Enviará a la salida estándar la concatenación de cada uno de los ficheros cuyos nombres h recibido como argumento, separando el contenido de cada fichero del contenido del siguiente mediante el separador de ficheros recibido. Al concluir devolverá 0.