**Parcial 1**

1) Suponga que es administrador de un servidor de correo y se le ha encargado un listado con todos los nombres de usuario que poseen cuenta de mail. Se sabe que un usuario que tiene cuenta de mail posee un directorio llamado mailDir en su directorio personal. Además de tener una cuenta de mail los usuarios del listado deben tener definido a bash como su shell por defecto. Para el listado solicitado se debe generar un archivo llamado listado.txt en el /var.

2) Se desea saber el uso del consumo de recursos del sistema por parte de los usuarios. Desarrolle un script que se correrá ingresando como argumento los nombres de los usuarios de los cuales se necesita obtener dicha información, no se sabe cuántos son con anticipación.

Para la realización del script debe definir las siguientes funciones:

* esta\_logueado(): devuelve true si el usuario pasado como argumento está logueado
* cant procesos(): devuelve la cantidad de procesos que está corriendo un usuario.
* uso\_de\_procesos(): devuelve true si el usuario pasado como argumento, usando la función cant\_procesos, está corriendo más de 100 procesos.

Utilizando la funcionalidad anterior, el script debe guardar en un archivo de log, en una ubicación apropiada del file system, el nombre de los usuarios que estén corriendo más de 100 procesos junto con la fecha de ejecución del script.

**Parcial 2**

1) Escriba un script que reciba una cantidad desconocida de parámetros al momento de su invocación (debe validar que al menos se reciba uno). Cada parámetro representa la ruta absoluta de un archivo o directorio en el sistema. El script deberá iterar por todos los parámetros recibidos, y:

* Si el parámetro es un archivo existente en el File System deberá comprimirlo dejando el archivo comprimido en lugar del original.
* Si el parámetro es un directorio existente en el File System:
  + Si tiene permiso de Lectura deberá empaquetarlo y comprimirio
  + Si tiene permiso de escritura deberá eliminarlo junto con todo su contenido
* Si el parámetro no es un elemento del File System (no existe), deberá contarlo e indicar la cantidad total al finalizar el script.

2) Realice un script que agregue todos los nombres de usuario del sistema a un arreglo e implemente las siguientes funciones:

* existe <parametro1>: Devuelve 0 si el usuario existe en el arreglo, 1 en caso contrario 2
* eliminar usuario <parametro1>: Si el usuario pasado como parámetro existe en el arreglo lo elimina del mismo. Caso contrario devuelve código de error 2
* usuarios\_con\_patron <parametro1>: Recorre el arreglo e imprime en pantalla los nombres de los usuarios en cuyos caracteres aparece el patrón pasado como parámetro
* cantidad: Imprime la cantidad total de usuarios en el arreglo
* usuarios: Imprime todos los nombres de los usuarios que están en el arreglo

**Parcial 3**

1) Escribir un script de Bash que reciba un nombre de usuario como parámetro. Si el usuario ingresado como parámetro NO es un usuario del sistema el script debe finalizar con el código de error 1. En caso contrario, debe verificar cada 30 segundos si el usuario está logeado. Al detectar que està logeado debe registrar en un archivo llamado access-<NOMRE DE USUARIO>.log el nombre del usuario junto con la fecha en que se lo detectó. NO debe borrar el contenido previo de este archivo y este archivo debe escribirlo en una ubicación apropiada del file system. Al detectar 30 veces que el usuario está logeado el script debe finalizar con el código de error 0.

2) Escribir un script de Bash que agregue a un arreglo los nombres de los archivos del directorio /var/log que tengan el texto "access" en su nombre y los agrege a un arreglo. Luego implemente las siguientes funciones sobre ese arreglo

* cantidad: Imprime la cantidad de archivos del directorio /var/log que tengan el texto "access" en su nombre
* listar: Imprime los nombres de todos los archivos del directorio /var/log que tengan el texto "access" en su nombre
* eliminar. Recibe como primer parámetro un indice del arreglo y como segundo parámetro la cadena "lógico" o "fisico". Si el segundo parámetro es "logico", solo borra la entrada en el arreglo, si es "fisico" borra la entrada en el arreglo y en el FileSystem. Aclaración: Para resolver las funciones debe valerse del arreglo

**Parcial 4**

1) Realice un script que implemente a través de la utilización de arreglos las siguientes funciones:

* insert elemento: La función insert recibe 1 parámetro (debe validar que así sea). Agrega al final del arreglo el elemento recibido como parámetro.
* rellenar n: Esta función itera n veces solicitando al usuario que ingrese un patrón de texto y lo agrega al final del arreglo. Debe validar que se reciba 1 parámetro
* select elemento: La función select recibe 1 parámetro (debe validar que así sea):
  + Si el parámetro es\*, entonces imprimirá todos los elementos del arreglo
  + Si al parámetro es distinto a \*, deberá verificar si en el arreglo existe un elemento que sea igual al parámetro. Si existe deberá imprimirlo en pantalla, caso contrario imprimirá "Elemento no encontrado"
* delete elemento: La función delete recibe 1 parámetro (debe validar que así sea):
  + Si el parámetro es\*, entonces eliminará todos los elementos del arreglo
  + Si al parámetro es distinto a \*, deberá verificar si en el arreglo existe un elemento que sea igual al parámetro. Si existe deberá eliminarlo, caso contrario imprimirá "Elemento no encontrado"

Además de las funciones debe proveer un ejemplo de uso de las funciones codificadas.

2) Realice un script que verifique cada 3 minutos si existe el archivo /var/log/any-service/error.log. Si el archivo existe y dentro de el hay al menos una linea con el texto "FATAL ERROR:" debemos: comprimir y empaquetar el contenido del directorio /var/log/usr-service, guardarlo en un directorio que es pasado como parámetro al script y además causar la terminación del script apropiadamente. Cuando el script termina debemos informar por la salida estandard la cantidad de lineas del archivo que contenian el texto "FATAL ERROR".

**Parcial 5**

1) Administrás un servidor que utiliza NGINX para publicar un conjunto de páginas web. Se te ha solicitado realizar un shell script con Bash para procesar su archivos de log, que reciba como argumentos una fecha y una o más direcciones IP, y que indique la cantidad de accesos que cada dirección IP ha realizado al servidor en la fecha que se recibe como primer argumento. La salida del script debe informar, utilizando dos columnas, la dirección IP y la cantidad de accesos que se registraron desde esa dirección IP.

Los archivos de log se ubican en el filesystem, en el directorio donde se guardan los logs de un servidor según el FHS, y sus nombres siguen el siguiente patrón: access.X.log donde X es la fecha del día en que se registraron los accesos. La estructura de esos archivos de log es la siguiente:

IP<ESPACIO>-<ESPACIO>FECHA<ESPACIO>"REQUEST"<ESPACIO>HTTP\_CODE<ESPACIO>CONTENT\_LEN

GTH ESPACIO>"USER\_AGENT"

Por ejemplo, esta es una línea de uno de esos archivos de log:

198.23.147.70 - 2023-05-03T06:28:41Z "GET / HTTP/1.1" 301 162 "Mozilla/5.0 (Windows NT 10.0; Win64; x64)

Chrome/99.107.0.0 Safari/537.36"

AppleWebKit/537.36 (KHTML, like Gecko)

Nota: tener en cuenta que el archivo de log para la fecha recibida como primer argumento puede no existir, en cuyo caso el script deberá informar el error y terminar con un código de salida adecuado.

2) Realizá un script de Bash que almacene en un arreglo los nombres de todos los usuarios del sistema que tengan un patrón, que se recibe como parámetro del script, en su nombre de usuario. Una vez almacenados, deberá presentar al usuario un menú que le permita realizar las siguientes operaciones sobre la lista obtenida:

* listar: lista en la salida estándar, de a uno por línea, los nombres de usuarios que se encuentran actualmente en el arreglo.
* eliminar: presenta la lista de usuarios con sus indices y luego le pide al usuario que indique el índice del elemento que se desee eliminar del arreglo.
* contar: presenta en la salida estándar la cantidad de elementos que actualmente tiene el arreglo. Nota: las operaciones deben implementarse como funciones que realicen las operaciones sobre el arreglo, validando los parámetros que recibe cuando fuera posible.

SOLO Para los que no contestaron autoevaluaciones:

3) Escribí un comando de una sola línea (podés usar pipes) que muestre todos los usuarios del sistema cuyo nombre de usuario contenga al menos una r en cualquier parte.

**Parcial 6**

1) Escribir un script de Bash que reciba un nombre de usuario como parámetro. Si el usuario ingresado como parámetro NO es un usuario del sistema el script debe finalizar con el código de error 1. En caso contrario, debe verificar cada 30 segundos si el usuario está logueado. Al detectar que está logueado debe registrar en un archivo llamado access-<NOMBRE DE USUARIO>Jog el nombre del usuario junto con la fecha en que se lo detectó. NO debe borrar el contenido previo de este archivo y este archivo debe escribirlo en una ubicación apropiada del file system. Al detectar 30 veces que el usuario está logueado el script debe finalizar retomando el código de error que indica que el proceso es exitoso.

2) Realice un script de Bash que agregue a un arreglo los nombres de los archivos del directorio /var/log que tengan el texto "access" en su nombre. Luego presente un menù al usuario con las siguientes funciones sobre ese arreglo

* cantidad: Imprime la cantidad de archivos del directorio /var/log que tengan el texto "access" en su nombre listar: Imprime los nombres de todos los archivos del directorio /var/log que tengan el texto "access" en su nombre eliminar. Solicita al usuario un indice del arreglo y un carácter que puede ser "T (lógico) o "T(fisico)". Si el segundo parámetro es "T", solo borra la entrada en el arreglo, si es "T" borra la entrada en el arreglo y el archivo del FileSystem.
* fin: Termina el script con el código de error que indica que el proceso es exitoso
* cantidad, listar y eliminar deben ser implementadas en funciones individuales

Ejercicio para alumnos que adeudan autoevaluaciones o no se inscribieron

3) Escriba un script de Bash que imprima en pantalla la cantidad de archivos del directorio /var/log/$SERVICE que contienen un patrón de texto que el usuario pasa como parámetro, $SERVICE es una variable de entomo, si la misma no posee ningún valor debe tomar el valor por defecto "local\_service". El script debe finalizar retornando el valor que indica que el proceso se ejecutó correctamente.