Diseña los siguientes scripts:

1) Que muestre de forma paginada el fichero de texto que le pasa como argumento de entrada.

2) Script que recibe un argumento y devuelve si es un fichero, un directorio o no existe. En el caso de que se trate de un fichero se especificara de que tipo es su contenido.

3) Script que recibe dos nombres de fichero como argumentos (comprobar que existen) y muestre por pantalla una comparativa en cuanto a número de líneas que tiene cada uno, número de palabras, número de caracteres y tamaño en bytes que ocupa.

4) Script que reciba dos argumentos, el nombre de un directorio y el nombre de un fichero a buscar. El script buscará si existe ese fichero en el directorio o en cualquiera de sus subdirectorios.

5) Un script que recibe n argumentos y los devuelva en orden inverso. Tened en cuenta que en la variable de shell BASH_ARGV que almacena los parámetros pasados el último argumento es el primer elemento del array (pila LIFO),así que no hay que darle la vuelta.

6) Script que sume los números que se pasen como parámetros.

7) Script que reciba tres argumentos: un número inicial, un número final y un incremento. El script mostrará por pantalla los números comprendidos entre el número inicial y el final en incrementos según indique el tercer parámetro.

8) Script encargado de dar de alta usuarios en el sistema. Al script no se le pasaran parámetros, preguntará interactivamente los siguientes datos: nombre de usuario, directorio HOME, grupo al que pertenece el usuario y login que se le asignará. El script comprobará que el login no está ocupado, que el grupo existe, creará el directorio HOME si no existe y asignará como shell /bin/bash

9) Script que toma como parámetro un directorio y indique si se tiene permisos o no para leer y acceder al contenido del directorio.

10) Script que recibe un directorio como primer parámetro, y a continuación una lista de archivos. El script debe validar que los parámetros recibidos sean realmente archivos y luego copiarlos al directorio recibido.

11) Un script que simule el funcionamiento del comando DIR de MSDOS. Se podrá incluir alguna de las opciones del comando DIR como: /s, /w y /p.

12) Script que simule el funcionamiento del comando DELTREE de MSDOS (elimina un directorio y todos los archivos y directorios que contiene). El comando solicitará confirmación por parte del usuario.

13) Script encargado de empaquetar uno o más directorios en un fichero. Recibirá como argumento el nombre del paquete que se pretende realizar (tar + gzip) y a continuación los directorios que se incluirán en ese paquete.

14) Descomprimir un paquete TGZ. Recibirá como argumento el nombre del fichero, comprobando su existencia.

15) Mostrar los diez primeros elementos de la tabla de multiplicar de un número introducido como parámetro.

16) Dado un número que se pasara como parámetro, indica si es o no divisible entre 101.

17) Script que puede recibir como parámetro la opción -h o el nombre de un archivo. Si recibe el nombre de un archivo, muestra si existe o no y de qué modo es accesible. Si recibe como parámetro -h, muestra por pantalla la sintaxis de ayuda para su ejecución.

18) Añadir cada 5 segundos una línea al archivo usuarios con la siguiente información: fecha | nº usuarios conectados en ese instante | nombres de usuarios (ver comandos **uniq**, **paste -s**). Ejemplo:

    Fri Mar 1 17:21:03 EDT 1996 | 3 | root tel21 sis12

    Fri Mar 1 17:21:03 EDT 1996 | 4 | tel21 so17 sis12 invitado

19) El script que recibira como parámetro el nombre de un proceso en ejecución y lo elimina, mostrando además el mensaje siguiente: El proceso nombre_proceso con PID PID_proceso_eliminado ha sido eliminado (ver comando **pidof**)

20) Script que permite adivinar al usuario cual es su UID. Pedira un número al usuario y cada vez que lo haga debe indicar al usuario si el UID es mayor o menor que el número introducido. Cuando se adivina el valor, se deben mostrar los intentos empleados.

21) Se recibirá como parámetros uno o dos nombres de archivo. Si recibe uno, debe permitir al usuario añadir texto a ese archivo de modo interactivo (leyendo una línea de la entrada estándar y añadiéndola al final del archivo, no con un editor) que se repetirá hasta que se teclee una palabra concreta Si son dos, añade al segundo todo el contenido del primero.

22) Script que toma como parámetros una serie de nombres de fichero de texto y muestra el nombre del fichero que tenga más líneas. Si hay más de un fichero con el número máximo de líneas debe mostrar el nombre de todos ellos. Ver comando **wc**, la opción **–l** de esta orden sirve para que muestre en la salida estándar el número de líneas del fichero que se pase como parámetro, junto con su nombre.

23) Cuando se crea una cuenta nueva en un sistema UNIX hay que buscar un UID para el nuevo usuario distinto a todos los existentes. Realiza un script que automatice esta tarea. El script debe tener en cuenta la información que aparece en el fichero ***/etc/passwd,*** y debe mostrar en la salida estándar el

primer entero mayor o igual a 0 que no esté siendo utilizado como UID. Sugerencia: para que su programa sea eficiente sería interesante que filtre la información almacenada en ***/etc/passwd,*** de cara a obtener una secuencia ordenada de los UIDs que están siendo utilizados. A partir de esta secuencia ordenada es sencillo obtener el dato buscado.

a. Realizar el mismo ejercicio anterior, teniendo en cuenta que una cuenta del sistema puede utilizar los UIDs 65534, 65565.

24) Script que espera hasta que un proceso “xclock” se ejecute. Y cuando ese proceso se ejecuta, lo mata (ver comando killall).

25) Realizar cuatro envíos al hacer ping (-c4) a una IP pasada como parámetro y indicar si la conexión ha tenido éxito o no (no se tendrá en cuenta si la ip es válida o no).

26) Mostrar un menú con los dispositivos usb, floppy y cd una vez elegido uno se pedirá si quiere montar o listar