PRÁCTICA 2:
Llamadas al sistema relacionadas con ficheros (II).

Objetivo:

Implementación de programas en C bajo UNIX, que utilicen las siguientes llamadas al sistema relacionadas con ficheros, además de las correspondientes a la Práctica 1:

link	***symlink***	***unlink***	***stat***
chmod	***access***	***umask***	

Desarrollo:

- Se implementarán los programas según el enunciado correspondiente a cada grupo de prácticas (A, B, C, D, E, F, G, H, I, J).

- Se seguirán los mismos criterios que en la Práctica 1, en cuanto a implementación del control de errores, mediante el uso de una función de error que muestre por pantalla el correspondiente mensaje de error (pudiendo hacer uso de la variable ***errno*** para indicar el error real ocurrido), y finalice la ejecución con una salida de error apropiada, que podrá ser conocida por un proceso padre mediante la variable **$?**.

- Además de los enunciados propuestos, el alumno puede plantearse mejoras en la implementación de los enunciados de la Práctica 1, utilizando las llamadas al sistema ***stat*** y ***access***, en cuanto a la comprobación de la existencia del fichero destino o el paso como parámetro en la línea de comandos de un directorio destino en lugar de un fichero destino (p. e. con *access (argv[2], 00)*).

Enunciados:

Grupo A

Implementar un programa al que se le pasen dos parámetros en la línea de comandos. El primer parámetro es una secuencia de caracteres, y el segundo, un nombre de fichero.

El programa buscará la secuencia de caracteres, especificada en el primer parámetro, en el fichero cuyo nombre ha sido especificado como segundo parámetro, de tal forma, que si encuentra una concordancia, muestra en pantalla la línea del fichero que contiene la secuencia especificada.

Grupo B

Implementar un programa al que se le pasen dos parámetros en la línea de comandos, correspondientes a dos supuestos ficheros con su ruta de acceso.

El programa contará el número de líneas del fichero pasado como primer parámetro, visualizando en pantalla el nombre del fichero y el número de líneas del mismo.

En caso de que el número de líneas sea inferior a 100, el programa creará un enlace duro, dado por el segundo parámetro introducido en la línea de comandos, que apunte al primer fichero.

Grupo C

Implementar un programa al que se pasen tres parámetros en la línea de comandos, correspondientes a tres nombres de fichero (por ejemplo, f1, f2 y f3). En primer lugar debemos comprobar que el fichero f1 existe. A continuación, debemos definir un enlace duro del fichero f2 al fichero f1 y un enlace simbólico del fichero f3 al fichero f1. Por último, mostraremos la información almacenada en el i-nodo del fichero f1 relativa al número de enlaces definidos sobre él.

Grupo D

Implementar un programa al que se le pase un parámetro en la línea de comandos, correspondiente a un nombre de fichero.

El programa, una vez comprobado que el fichero existe, procederá al borrado del mismo mediante un proceso interactivo con el usuario para pedir su conformidad. En caso de que se confirme la operación de borrado del fichero, se borrará el fichero, independientemente del número de enlaces duros que haya definidos sobre él. Si se da la circunstancia de que hay más de un enlace duro asociados al fichero, el programa los borrará para luego poder borrar el fichero físicamente.

Grupo E

Implementar un programa al que se le pase un parámetro en la línea de comandos, correspondiente a un supuesto fichero con su ruta de acceso.

El programa contará el número de caracteres del fichero pasado como parámetro, visualizando en pantalla el nombre del fichero y el número de caracteres del mismo. Luego, comprobará que el número de caracteres contados coincide con el tamaño en bytes obtenido

con la llamada *stat* (por ejemplo, visualizando en pantalla el tamaño del fichero a partir de la información devuelta por dicha llamada al sistema).

Grupo F

Implementar un programa que dado un número indeterminado de ficheros pasados como parámetros en la línea de comandos, muestre en pantalla la siguiente información **sólo para los ficheros existentes**:

- Nombre.
- Tamaño en bytes.
- Número de enlaces definidos sobre el ficheros.
- Identificación del usuario.

Grupo G

Implementar un programa que dado un número indeterminado de ficheros pasados como parámetros en la línea de comandos, muestre en pantalla la siguiente información **sólo para los ficheros existentes**:

- Nombre.
- Tamaño en bytes.
- Número de enlaces definidos sobre el ficheros.

Además, todo fichero existente sobre el que tengamos derechos de escritura y sobre el que haya definido únicamente un enlace (el de creación), debe ser borrado.

Grupo H

Implementar un programa que dado un número indeterminado de ficheros pasados como parámetros en la línea de comandos, muestre en pantalla la siguiente información **sólo para los ficheros existentes**:

- Nombre.
- Tamaño en bytes.

Además, todo fichero existente sobre el que tengamos derechos de escritura debe ser borrado, independientemente del número de enlaces que haya definidos sobre él.

Grupo I

Implementar un programa al que se le pase un parámetro en la línea de comandos, correspondiente a un nombre de fichero sin más enlaces asociados que el de creación.

El programa, una vez comprobado que el fichero existe, procederá al borrado del mismo mediante un proceso interactivo con el usuario para pedir su conformidad. En caso de que se confirme la operación de borrado del fichero, se comprobará en primer lugar si hay permiso de escritura sobre ese fichero. Si tenemos permiso borraremos el fichero directamente, y si no cambiaremos los bits de protección adecuados antes de borrar el fichero (suponemos que el fichero con el que trabajamos ha sido creado por nosotros).

Grupo J

Implementar un programa que dado un número indeterminado de ficheros pasados como parámetros en la línea de comandos, muestre en pantalla la siguiente información **sólo para los ficheros existentes**:

- Nombre del fichero.
- Tipo de fichero.
- Tamaño en bytes.
- Número de enlaces del fichero.
- Identificación del usuario.

En caso de que se trate de un fichero regular, se cambiarán los permisos definidos sobre el fichero, asignando permiso de lectura y escritura a todo el que acceda al sistema (suponemos que los ficheros con los que trabajamos han sido creados por nosotros).