Tarea Extraclase

De

Sistema Operativo.

Título: ¨Comando para Minix 3¨.

Integrante:

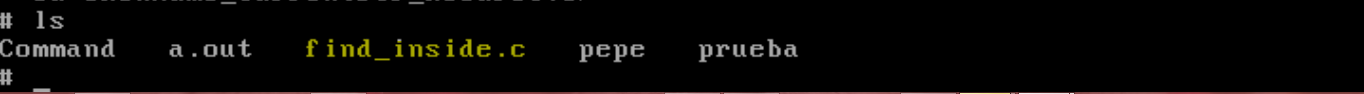
* Luis Ernesto Glez. Cárdenas.

Introducción.

En la asignatura de Sistema Operativo, hemos desarrollado habilidades de modificar y agregar funcionalidades al Sistema Operativo Minix 3. En el presente trabajo presentaremos como desarrollar un comando para el sistema. El comando que crearemos fue creado con fines educativos y con objetivos futuros a modificaciones, por lo que es válido aclarar, que una mejor solución hubiese sido haberlo programado en Perl (Lenguaje estudiado en la asignatura previamente dicha). El comando consiste en dado un directorio, hacer una búsqueda, primero en profundidad y cada vez que encuentre un archivo, lo abrimos y buscamos en él un patrón dado, si se encuentra ese patrón devolvemos el camino hacia ese archivo.

Desarrollo

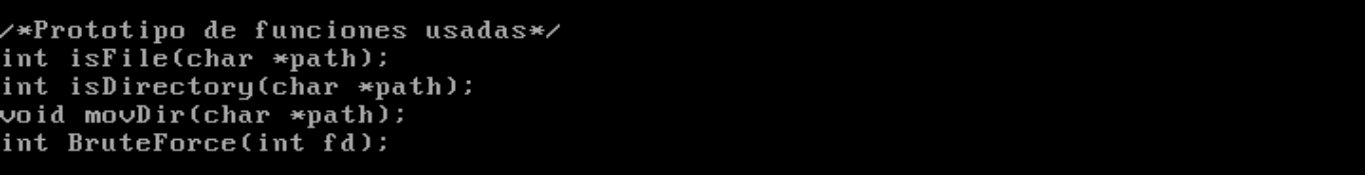
Para desarrollar el Comando, primeramente debemos crear el archivo donde implementaremos nuestro código, para ello crearemos el archivo find\_inside.c. Como muestra la figura(1).



Fig(1)

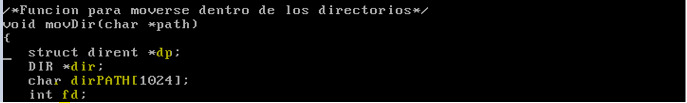
Como nos vamos a mover por directorios y buscar en archivos, debemos tener una buena ética de programación por lo que vamos a necesitar las siguientes funciones:

* void movDir(char \*path) ---> Función usada para movernos por los directories.
* int BruteForce(int fd) ---> Función usada para buscar dentro de un archive el patron especificado, usando el algoritmo de fuerza bruta, la cual queda abierta para mejoras futuras.
* int isDirectory(char \*path) ----> Función usada para saber si es un directorio o no.
* int isFile(char \*path)---->Función usada para saber si es un archivo.



Fig(2):Prototipo

Cada función tiene sus detalles y uso de estructuras que ya están implementadas, por ejemplo la función movDir:

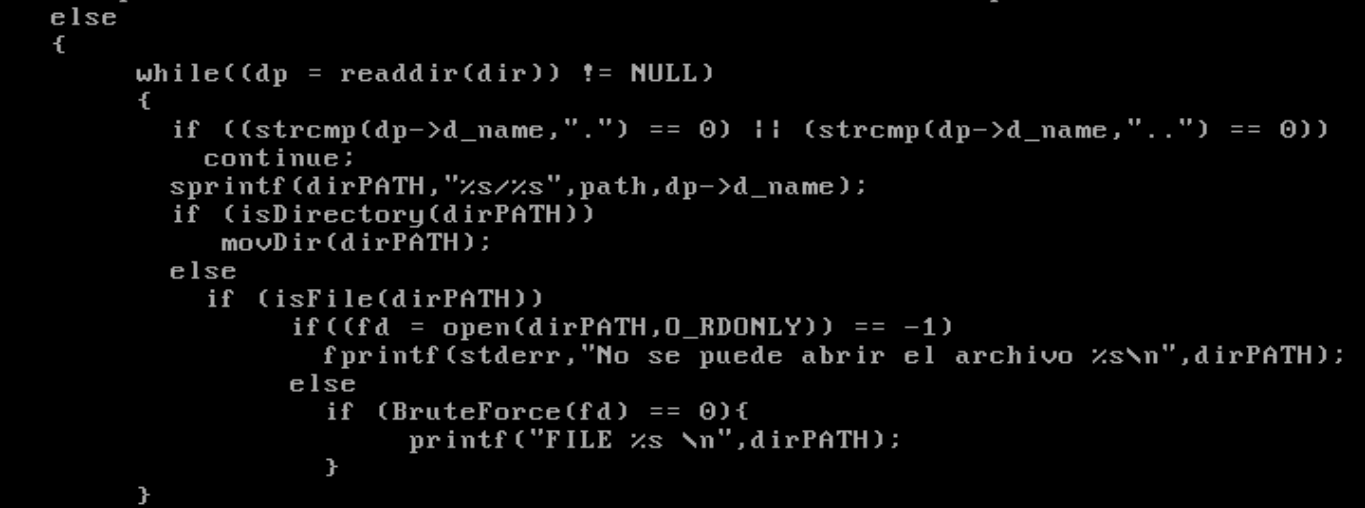


Fig(3):Variables

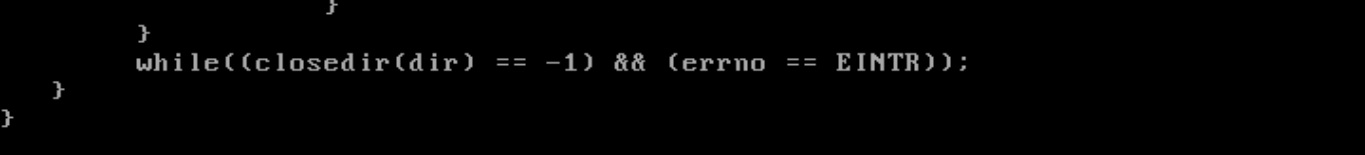
La estructura dp va a contener el nombre y el numero de i-node del archivo, dir es una secuencia de entradas a un directorio y dirPATH es la usada para salvar el camino completo hacia los subdirectorios y sus archivos.

Fig(4):Opendir

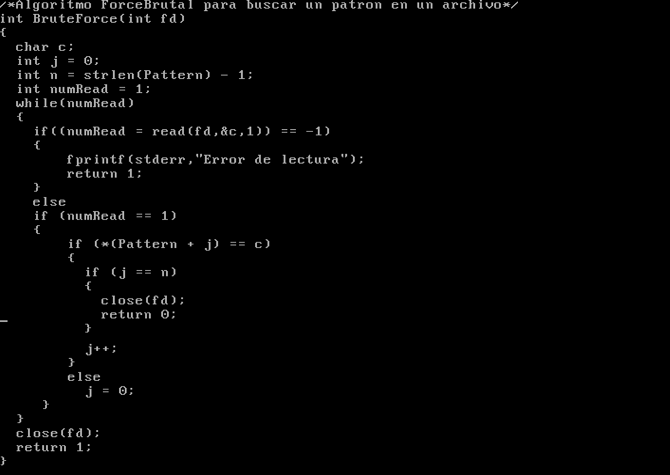
Aquí dir va a tomar la secuencia de entrada al directorio path, pero se devuelve NULL, no devolvemos error pues recuerde que estamos buscando por un árbol de directorio y puede ser que no tengamos permisos para entrar en un directorio dado.

Fig(5): Listar directorio

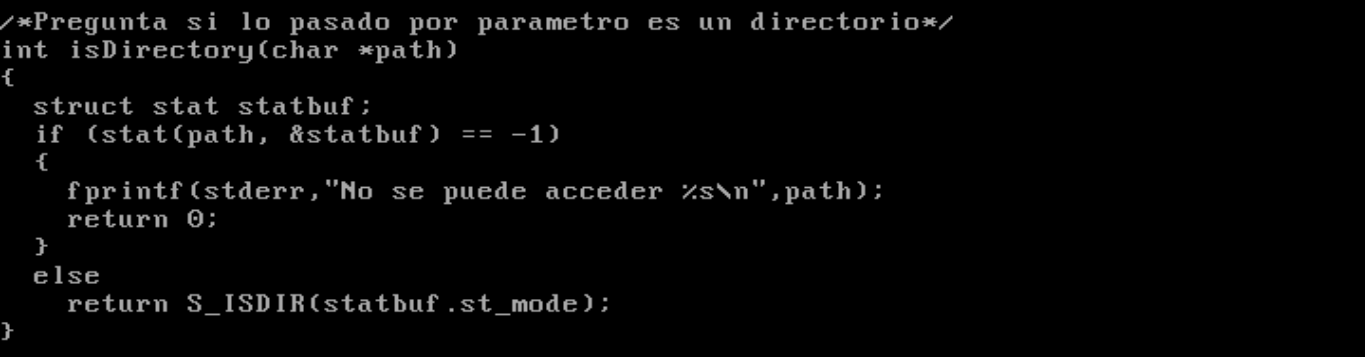
Vamos a ir leyendo de dir hasta que sea NULL. Fijémonos que si el directorio es ¨.¨ o ¨..¨ nunca entramos, pues si entráramos estaríamos haciendo una recursividad infinita. En esta parte del código de movDir es en la que preguntamos que si isDirectory es mayor que 0, significa que es un directorio y que podemos llamar la función movDir de nuevo, pero ahora con ese nuevo directorio como parámetro, pero antes tener en cuanta que tenemos que concatenar el camino anterior. Y si es un archivo llamamos entonces al método BruteForce qu buscará el patrón en el archivo.

Fig(6):Close Dir

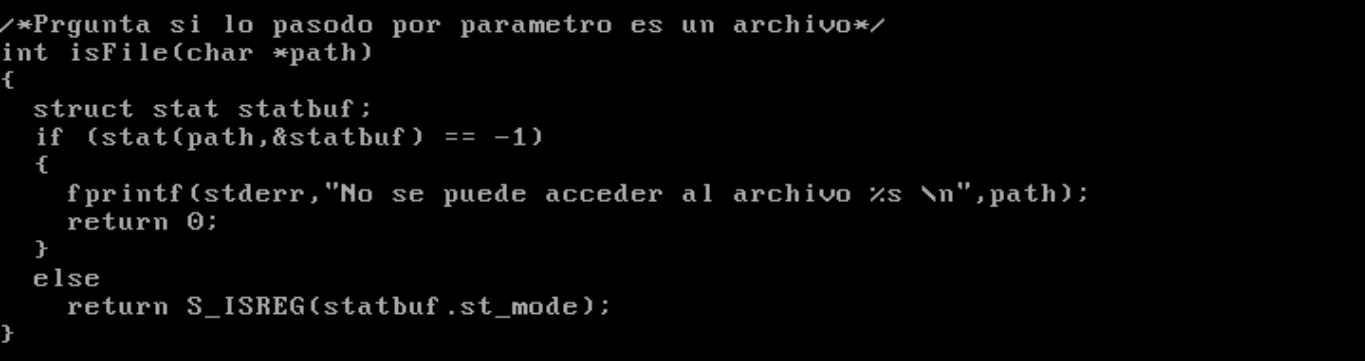
Como hemos estado abriendo directorios recordemos cerrarlos todos

Fig(7):BruteForce

El algoritmo de BruteForce es algo sencillo, pues el cual busca en un archivo un archivo, un patrón determinado. Devuelve 0 si lo encuentra y 1 si no lo encuentra.

Fig(8)isDirectory

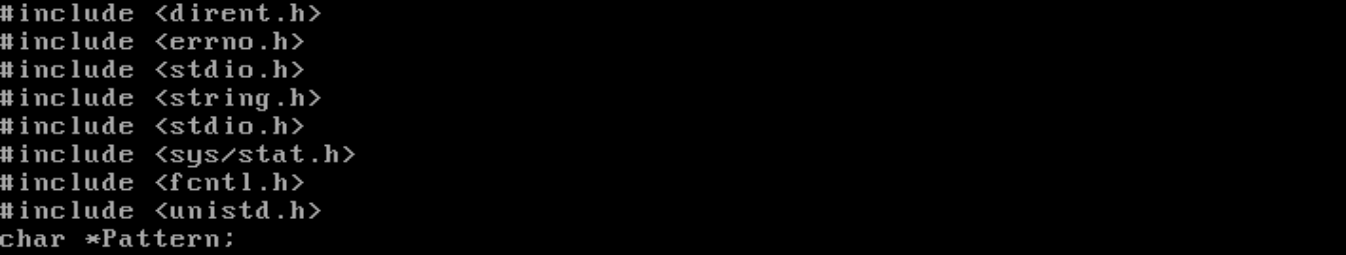
En esta función usamos statbuf que va a contener todas las propiedades del archivo, las cuales se van a inicializar con la llamada al sistema stat( … ). También usamos la MACRO S\_ISDIR que la usamos para ver si es un directorio.



Fig(9):isFile

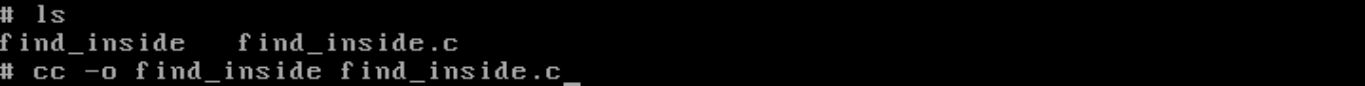
Esta función es muy parecida a la anterior con la diferencia que aquí la MACRO que se usa es S\_ISREG para ver si es un archivo regular.

Debemos recordar que para poder usar muchas de estas funciones es necesario implementar las directivas al procesador necesarias.



Fig(10): Directivas al procesador

Por ultimo debemos compilar con la siguiente instrucción y generar el ejecutable, el cual debemos copiar en cualquier camino que contenga la variable PATH.



Conclusiones

Hemos de desarrollado un nuevo Comando para el Sistema Operativo Minix 3, con el cual nos hemos familiarizado mas con dicho sistema, además de profundizar en el lenguaje C. Esperemos poder modificar el comando en el futuro.

Bibliografía:

Operating Systems - Design and Implementation, 3rd Edition By Andrew S. Tanenbaum.

Unix Systems Programming By Kay A. Robbins, Steven Robbins.

The C programming Language By Brian W. Kernighan and Dennis M. Ritchie.