Sistemas Operativos - Manejo de archivos

----------------------------------------

1) Escriba un programa que muestre en orden todos los argumentos

pasados por linea de comando. Para ello argumente la funcion main()

de su programa de la siguiente manera

int main(int argc, char \*argv[])

De esta manera "argc" contiene la cantidad de elementos de la linea

de comando, mientras que "argv[]" es un array de char que contiene

cada uno de los componentes.

Tip: Solo debe recorrer el arreglo argv imprimiendo su contenidox|

for ( i = 0; i < argc; i++ ) {

... argv[i] ...

}

2) Genere un archivo que contenga un listado extendido del contenido

del directorio raiz (/). Escriba un programa que lea dicho archivo

y muestre su contenido en pantalla. Para ello use las llamadas

al sistema open(), close() y read().

Tip: Para crear el archivo basta con hacer un 'ls -l' y redirigir

su salida.

Recuerde que cada proceso 'nace' con entrada estandar, salida estandar

y error estandar.

'<' redirige entrada std

'>' redirige salida std, igual que '1>'

'2>' redirige error std

Tip: Para su programa debe:

a) abrir el archivo para lectura utilizando open()

b) recorrer el archivo haciendo read() y observando su

valor de retorno, el mismo nos indica cuando hemos llegado

al final del archivo.

c) cerrar el archivo

Consejo: Siempre verifique valores de retorno de funciones (ver man page)

ej.

fd = open( "test.txt", O\_RDONLY ); /\* verificar el valor de fd \*/

n = read( fd, buff, max ); /\* verificar n, su valor es importante \*/

3) Escriba un programa que lea el directorio raiz (/) y muestre su

contenido por pantalla. Use opendir(), readdir() y closedir().

6) Escriba un programa que reciba por linea de comando el nombre de

un archivo y lo copie en un archivo con la extension ".bak". Use

open(), close(), read() y write().

Tip: Usará open() dos veces, una el leer el archivo origen y otra

al crear y escribir en el archivo de destino. Para el segundo caso

puede utilizar

fd = open("test.txt", O\_RDWR|O\_CREAT, 0644);

o lo que es lo mismo

fd = open("test.txt", O\_RDWR|O\_CREAT, S\_IRUSR|S\_IWUSR|S\_IRGRP|S\_IROTH );

vea la página de manual 'man 2 open'

Tip: El programa en si tiene tres partes

- abrir los archivos

- recorrer (loop) el origen y escribir en destino

- cerrar los archivos

el loop termina analizando el valor retornado por read()

7) Escriba un programa que reciba por linea de comando el nombre de

un archivo, a continuacion muestre toda la informacion que se

pueda obtener de ese archivo usando la funcion "stat()".

AYUDA: Lo que hay que mostrar en este caso son los distintos

componentes de la estructura devuelta por la funcion stat().

struct stat

{

dev\_t st\_dev; /\* dispositivo \*/

ino\_t st\_ino; /\* inodo \*/

mode\_t st\_mode; /\* protecci� \*/

nlink\_t st\_nlink; /\* nmero de enlaces f�icos \*/

uid\_t st\_uid; /\* ID del usuario propietario \*/

gid\_t st\_gid; /\* ID del grupo propietario \*/

dev\_t st\_rdev; /\* tipo dispositivo (si es

dispositivo inodo) \*/

off\_t st\_size; /\* tama� total, en bytes \*/

unsigned long st\_blksize; /\* tama� de bloque para el

sistema de ficheros de E/S \*/

unsigned long st\_blocks; /\* nmero de bloques asignados \*/

time\_t st\_atime; /\* hora ltimo acceso \*/

time\_t st\_mtime; /\* hora ltima modificaci� \*/

time\_t st\_ctime; /\* hora ltimo cambio \*/

};

De estos campos (man 2 stat) podemos mostrar como enteros (%d) a:

st\_ino, st\_nlink, st\_uid, st\_gid, st\_size, st\_blksize, st\_blocks.

Las fechas deben ser pasadas a la funcion ctime() que retorna una

cadena, Ej. ctime(&var.st\_atime) e imprimir esto como cadena (%s).

Para el manejo de el campo st\_mode, debemos hacer uso de lo

siguiente:

a) Para imprimir los permisos iterpretamos ese campo como una

representacion octal, con lo cual lo imprimos

("%#o", var.st\_mode & ~(S\_IMFT));

b) Para descubrir cual es el tipo de archivo usamos los macros

definidos en man 2 stat, a saber por ej. S\_ISLNK, de modo que

podemos escribir

if (S\_ISLNK(var.st\_mode)) {

printf("Es un vinculo simbolico"):

}

y asi sucesivamente para el resto de los macros.

c) Por ultimo, para interpretar el campo st\_dev, debemos agregar

un archivo de cabecera

#include <linux/fs.h>

En este archivo estan incluidos dos macros MAJOR(m) y MINOR(m) a

los cuales les damos como argumento el campo st\_dev y nos

retornara los numeros mayor y menor correspondientes al dipositivo

fisico, para ver esto pueden listar el dispositivo correspondiento

al disco duro

ls -l /dev/sda6

y al observar el campo de tamaño se ven dos numeros, estos son el

numero mayor y el menor que corresponden al dispositivo en este

caso

mayor 8 -> SCSI8\_MAYOR

minor 6 -> particion 6 (en realidad disco logico)

Ambos numeros se pueden mostrar como enteros.