ARCHIVOS Y DIRECTORIOS EN GNU/LINUX.

Manejo de archivos en GNU/Linux

Como se explicó en las clases teóricas, el manejo de archivos se lleva a cabo mediante llamadas POSIX de manejo de archivos.

Estas funciones (en FreePascal) se encuentran en la unit BaseUnix:

fpOpen	Abre/crea un descriptor de archivo.
fpRead	Lee datos desde un descriptor de archivo.
fpWrite	Escribe en un descriptor de archivo.
fpLSeek	Mueve el puntero de posición de un archivo.
fpClose	Cierra un descriptor de archivo.

Para ver otras funciones de manejo de archivos vea la documentación de FreePascal.

Ingrese al sistema como **usuario no privilegiado**. Cree un directorio donde almacenar los programas de este práctico dentro de su directorio home. Luego sitúese en el directorio recién creado.

Escriba el siguiente programa, nómbrelo "**prog_0701.pp**" y compílelo.

```
Program prog_0701;

uses BaseUnix;

Var fd : Longint;
    datos: String;

begin
  fd := FPOpen(ParamStr(1),O_WrOnly OR O_Creat);
  if fd > 0 then
    begin
    Writeln('Escriba lo que desea en el archivo:');
    Readln(datos);
      if (Length(datos)+1) <> (FPWrite(fd,datos,Length(datos)+1)) then
       Writeln ('Error al escribir en el archivo!!!');
    FPClose(fd);
    end;
end.
```

Cree un archivo vacío. Ejecute el programa pasándole como parámetro el archivo recién creado e ingrese algo de texto (al menos 20 caracteres) para escribir dentro del archivo. Verifique que la información se grabó en el archivo.
(1) Ejecute el programa nuevamente pasándole como parámetro un archivo inexistente. Se ejecutó igual? Grabó la información dentro del archivo? Por qué?
(2) Ejecute el programa nuevamente pasándole como parámetro el primer archivo e ingrese otra información (al menos 20 caracteres) para grabar en el archivo. Verifique que la información se grabó en el archivo. Qué ocurrió con la información que habíamos grabado en la primera ocasión? Por qué?

La documentación de FreePascal (rtl.pdf) pueden ser de gran ayuda para responder a las preguntas anteriores.

Escriba el siguiente programa y nómbrelo "**prog_0702.pp**" y compílelo.

```
Program prog_0702;

uses BaseUnix;

var fd : Longint;
    datos: String;

begin
  fd := fpOpen(ParamStr(1),O_RdOnly);
  if fd>0 then
    begin
        if fpRead(fd,datos,10) < 0 then
          begin
          Writeln ('Error leyendo archivo!!!');
          Halt(2);
          end;
      Writeln(datos);
  end;
  fpClose(FD);
end.
```

(3) Ejecute el programa pasándole como parámetro el archivo de texto del ejercicio anterior. Observe los resultados. Por qué no mostró la totalidad del archivo?

Manejo de archivos y directorios en GNU/Linux

En un nivel más bajo se puede acceder mediante funciones POSIX, a operaciones generales de manejo de archivos, directorios y sistemas de archivos.

Estas funciones (en FreePascal) se encuentran en la unit BaseUnix:

fpAccess	Chequea los permisios de acceso a un archivo
fpChown	Cambia el dueño de un archivo.
fpChmod	Cambia los permisos de acceso de un archivo.
fpStat	Brinda información sobre un archivo.
fpRename	Renombra un archivo.
fpFStat	Brinda información sobre el filesystem.
fpLStat	Brinda información sobre un enlace (link).
fpLink	Crea un enlace (link).
fpReadLink	Lee el contenido de un enlace simbólico.
fpSymLink	Crea un enlace simbólico.
fpUnLink	Elimina un archivo.
fpUtime	Cambia la fecha de un archivo.
fpOpenDir	Abre un directorio para lectura.
fpCloseDir	Cierra un directorio,
fpReadDir	Lee una entrada de directorio.

Y éstas en la unit Unix:

FLock	Bloquea un archivo.
TellDir	Indica la entrada de directorio actual.
SeekDir	Se ubica en una entrada determinada del directorio.

Para ver otras funciones de manejo de archivos, directorios y sistemas de archivos vea la documentación de Freepascal.

(4) Utilizando la documentación de Freepascal emplee la función **fpAccess** para reescribir los programas anteriores de manera que verifiquen que el archivo existe y que el usuario tiene los privilegios para realizar las operaciones sobre el mismo.

Escriba el siguiente programa y nómbrelo "**prog_0703.pp**" y compílelo.

```
program prog_0703;
uses BaseUnix;
var info : Stat;
begin
  if FPStat(ParamStr(1),info)<>0 then begin
    Writeln('Fallo la llamada a fpFSat!!!');
     halt (1);
  end;
  writeln ('Resultados del fpFstat del archivo:');
  writeln ('I-nodo                    : ',info.st_ino);
  writeln ('Modo (Tipo y Permisos)    : ',info.st_mode);
  writeln ('Numero de links           : ',info.st_nlink);
  writeln ('User ID del duenio        : ',info.st_uid);
  writeln ('Group ID del duenio       : ',info.st_gid);
  writeln ('Tipo de dispositivo inodo : ',info.st_rdev);
  writeln ('Tamanio del Bloque        : ',info.st_blksize);
  writeln ('Tamanio en Bytes          : ',info.st_size);
  writeln ('Nro. de Bloques           : ',info.st_blocks);
  writeln ('Ultimo acceso             : ',info.st_atime);
  writeln ('Ultima modificacion       : ',info.st_mtime);
  writeln ('Ultimo cambio             : ',info.st_ctime);
end.
```

(5) Pruebe el programa pasándole como parámetro distintos archivos. Dónde está almacenada esta información de cada archivo?
Puede obtener más información sobre el registro Stat mediante la pagina man del mismo Stat(2).

Escriba el siguiente programa y nómbrelo "**prog_0704.pp**" y compílelo.

```
program prog_0704;
uses BaseUnix;
var info : Stat;
begin
  if fpLStat(ParamStr(1),info)<>0 then begin
    Writeln('Fallo la llamada a fpLStat!!!');
     halt (1);
  end;
  writeln ('Resultados del fpLStat del archivo:');
  writeln ('I-nodo                    : ',info.st_ino);
  writeln ('Modo (Tipo y Permisos)    : ',info.st_mode);
  writeln ('Numero de links           : ',info.st_nlink);
  writeln ('User ID del duenio        : ',info.st_uid);
  writeln ('Group ID del duenio       : ',info.st_gid);
  writeln ('Tipo de dispositivo inodo : ',info.st_rdev);
  writeln ('Tamanio del Bloque        : ',info.st_blksize);
  writeln ('Tamanio en Bytes          : ',info.st_size);
  writeln ('Nro. de Bloques           : ',info.st_blocks);
  writeln ('Ultimo acceso             : ',info.st_atime);
  writeln ('Ultima modificacion       : ',info.st_mtime);
  writeln ('Ultimo cambio             : ',info.st_ctime);
end.
```

Cree un archivo de texto llamado **prueba.txt** y agreguele algo de contenido.
Ejecute la siguiente línea:

```
ln -s prueba.txt prueba1.txt
```

Luego ejecute:

```
./prog_0703 prueba.txt
./prog_0703 prueba1.txt
./prog_0704 prueba1.txt
```

(6) A qué se deben las diferencias y similtudes en las tres salidas? (preste especial atención a los campos i-nodo y tamaño)
Escriba el siguiente programa y nómbrelo "**prog_0705.pp**" y compílelo.

```
program prog_0705;

uses BaseUnix;
var archivo: Stat;
begin
  if fpStat(ParamStr(1),archivo)=0 then
    begin
    if fpS_ISLNK(archivo.st_mode) then
      Writeln ('El archivo es un enlace...');
    if fpS_ISREG(archivo.st_mode) then
      Writeln ('El archivo es un archivo regular...');
    if fpS_ISDIR(archivo.st_mode) then
      Writeln ('El archivo es un directorio...');
    if fpS_ISCHR(archivo.st_mode) then
      Writeln ('El archivo es un dispositivo de caract...');
    if fpS_ISBLK(archivo.st_mode) then
      Writeln ('El archivo es un dispositivo de bloques...');
    if fpS_ISFIFO(archivo.st_mode) then
      Writeln ('El archivo es una canieria (Pipe)...');
    if fpS_ISSOCK(archivo.st_mode) then
      Writeln ('El archivo es un socket...');
  end;
end.
```

(7) Pruebe el programa pasándole como parámetro archivos de distintos tipos (enumerelos y diga cuál fue el resultado). Dónde está almacenada esta información de cada archivo?

Escriba el siguiente programa y nómbrelo "**prog_0706.pp**" y compílelo.

```
program prog_0706;
uses BaseUnix;
var directorio : PDir;
    entrada : PDirent;

begin
  directorio := fpOpenDir(ParamStr(1));
  repeat
    entrada := fpReadDir(directorio^);
    if entrada <> nil then
    With entrada^ do
      begin
          Writeln ('-----------------------------------');
{      Writeln ('Entrada No    : ',Entry);} {puede ser que no funcione}
          Writeln ('Nombre        : ',pchar(@d_name[0]));
          Writeln ('I-nodo        : ',d_fileno);
          Writeln ('Offset        : ',d_off);
          Writeln ('Long. Nombre   : ',d_reclen);
      end;
  until entrada = nil;

  fpCloseDir (directorio^);
end.
```

Pruebe el programa pasándole como parámetro distintos directorios.

Escriba el siguiente programa y nómbrelo "**prog_0707.pp**" y compílelo.

```
program prog_0707;
uses BaseUnix, Unix, Unixtype;
var ubicacion : pchar;
    fd: cint;
    fsinfo : TStatFS;
begin
  ubicacion:='.';
  fd:= fpOpen(ubicacion, O_RdOnly);
  if fstatfs(fd,fsinfo)<>0 then
       begin
       Writeln('Fallo el fpStatFS. Error No : ',fpgeterrno);
       Halt(1);
       end;
    Writeln ('Tipo de FS           : ',fsinfo.fstype);
    Writeln ('Tamanio de bloque    : ',fsinfo.bsize);
    Writeln ('Bloques libres       : ',fsinfo.bfree);
    Writeln ('Bloques disponibles  : ',fsinfo.bavail);
    Writeln ('Archivos             : ',fsinfo.files);
    Writeln ('Descriptores libres  : ',fsinfo.ffree);
    Writeln ('Identificacion de FS : ',fsinfo.fsid[0]);
    Writeln ('Long. del Nombre     : ',fsinfo.namelen);
end.
```

(8) Ejecute el programa. Ubíquese en el directorio raíz (**/**) y vuelva a ejecutar el programa. Ahora sitúese en el directorio **/proc** y ejecute nuevamente el programa. Monte un pendrive y ubíquese en un directorio del mismo y vuelva a correr el programa. Sobre qué filesystems nos está mostrando información en cada ocasión? Explique el significado de cada resultado.

Puede obtener más información sobre el registro StatFS mediante la pagina man del mismo StatFS(2).

(9) Modifique el programa anterior (nómbrelo **"prog_0708.pp"**) para que muestre el espacio total, ocupado y libre (en bytes) en el sistema de archivos.

(10) Realice su propia versión del comando **ls** empleando las llamadas POSIX de gestión de archivos y directorios. Nómbrelo **"prog_0709.pp"**, compílelo y pruebelo listando distintos directorios. Para mantener la sencillez, el comando deberá cumplir con las siguientes características:

- Deberá mostrar (en este orden): permisos del usuario actual (**rwx**), tamaño en Bytes y nombre de cada archivo del directorio en cuestión. (Uno por línea)
- Deberá ademas, permitir diferenciar archivos de ciertos tipos: ejecutables (por el usuario actual), enlaces simbólicos, directorios y dispositivos.
- Deberá mostrar al final el número de archivos listados.
- Deberá mostrar mensajes significativos en caso de no poder realizar el listado.

Confeccione un informe **original**, **conciso** y **completo** donde se dé respuesta a las preguntas y consignas precedidas por un número encerrado entre paréntesis. Éste informe deberá ser confeccionado y entregado por cada grupo que llevó a cabo las actividades.
La longitud máxima del informe es de dos páginas (sin contar las líneas correspondientes a código fuente).