Operating Systems

Grado en Informática. Course 2021-2022

Lab Assigment 1: File systems. In this lab assigment we'll do the following task

- we'll add commands to the shell started in the previous lab assignment, so that it can view and manipulate the filesystem, listing, adding and deleting files or directories.

Add to the shell, started in previous assignments, the following commands

crear [-f] name Creates a file or directory in the file system. **name** is the name of the file (or directory) to be created. If **-f** is specified an empty file is to be created, otherwise a directory will be created. If *name* is not given, the name of the current working directory will be printed Example:

```
->crear -f fich.txt
->crear carpeta
->crear /root/folder
cannot create /root/folder: permission denied
->crear
/home/antonio/c
```

borrar name1 name2 ... Deletes files and/or empty directories

borrarrec name1 name2 ... Deletes files and/or non empty directories.

- *borrar* deletes files and/or empty directories. name1, name2 ... represent the files or directories to be deleted,
- *borrarrec* deletes files and/or directories together with ALL OFF THEIR CONTENT.
- If no *name* is given, the name current working directory will be printed (as with the *carpeta* command)
- When a file or directory cannot be removed, an appropiate message must be given to the user

listfich [-long] [-link] [-acc] name1 name2 name3 ... Gives info on files (or directories, or devices ...) name1, name2 ... in ONE LINE per file.

If no options are given, it prints the size and the name of each file. If no *name* is given, the name current working directory will be printed (as with the *carpeta* command)

- **-long** stands for long listing, it will print out the date of last modification (in format YYYY/MM/DD-HH:mm), number of links, owner, group, mode (drwx format), size and name of the file. If any of the names is a directory, information on the directory file itsself will be printed. The format to be used is:

  ```
  date number_of_links (inode_number) owner group mode size name
  ```

- **-link** is only meaningful for long listings: if the file is a symbolic link the name of the file it points to is also printed

  ```
  date number_of_links (inode_number) owner group mode size name->file_the_link_points_to
  ```

- **-acc** last access time will be used instead of last modification time

listdir [**-reca**] [**-recb**] [**-hid**] [**-long**] [**-link**] [**-acc**] **name1 name2 . . .** Lists **the contents** of directories with names *name1, name2 . . . .* If any of name1, name2 . . . is not a directory, info on it will be printed **EXACTLY** as with command *listfich*. If no *name* is given, the name of the current working directory will be printed (as with the *carpeta* command)

- **-long, -link and -acc** Have exactly the same meaning as in listfich, but affect all files inside directories. The format in which a file in a directory is going to be listed must be **EXACTLY** the same as with *listfich*.
- **-hid** hidden files and/or directories (those whose name starts with **.**) will also get listed.
- **-reca** means that, when listing a directory's contents, subdirectories will be listed recursively AFTER all the files in the directory. (if the *-hid* option is also given, hidden subdirectories will also get listed, except **.** and **..** to avoid infinite recursion).
- **-recb** means that, when listing a directory's contents, subdirectories will be listed recursively BEFORE the directory they are in. (if the *-hid* option is also given, hidden subdirectories will also get listed, except **.** and **..** to avoid infinite recursion).
- if not *name* is given, the current working directory will be printed (as with the *carpeta* command).

- To check what type of filesystem object a name is, one of the *stat* system calls must be used. **DO NOT USE THE FIELD `d_type` IN THE DIRECTORY ENTRY**.
- Note that the *-long -link* and *-acc* option also affect listing the contents of directories.

Example:

```
-> listffich -long   enlace p1.c /home/antonio fjhskfhsdkf
2021/09/15-13:35    (1) 15889016 antonio antonio -rw-r--r--       4  enlace
2021/09/10-11:40    (1) 15888887 antonio antonio -rw-r--r--    2713  p1.c
2021/05/04-12:13  (156) 15859713 antonio antonio  drwxrwxrwx  12288  /home/antonio/
cannot access fjhskfhsdkf: No such file or directory
->listfich -long  -link   enlace p1.c
2021/09/15-13:35    (1) 15889016 antonio antonio -rw-r--r--       4  enlace->p1.c
2021/09/10-11:40    (1) 15888887 antonio antonio -rw-r--r--    2713  p1.c
```

Information on the system calls and library functions needed to code these programas is available through `man`: (*open, opendir, readdir, lstat, unlink, rmdir, realpath, readlink . . .)*.

IMPORTANT:

- These programs should compile cleanly (produce no warnings even when compiling with `gcc -Wall`)
- **NO RUNTIME ERROR WILL BE ALLOWED (segmentation, bus error . . .)**. Programs with runtime errors will yield no score.
- These programs can have no memory leaks
- When one program cannot perform its task (for whatever reason, for example, lack of privileges) it should inform the user (See *errors* section at *HEPFUL INFORMATION*)
- All input and output is done through the standard input and output

HELPFUL INFORMATION

The following funtions (*ConvierteModo(), ComvierteModo2()* and *ConvierteModo3()*) convert the mode of one file (in a *mode_t* integer) to `"-rwxrwxrwx"` form. Note that these three functions have different ways of memory allocation. You can use any of them (it's the programmer's choice), but make sure you understand how they work and the implications of choosing one over the others

```
char LetraTF (mode_t m)
{
  switch (m&S_IFMT) { /*and bit a bit con los bits de formato,0170000 */
    case S_IFSOCK:  return 's';  /*socket */
    case S_IFLNK:   return 'l';    /*symbolic link*/
    case S_IFREG:   return '-';   /* fichero normal*/
    case S_IFBLK:   return 'b';   /*block device*/
    case S_IFDIR:   return 'd';   /*directorio */
    case S_IFCHR:   return 'c';   /*char  device*/
    case S_IFIFO:   return 'p';   /*pipe*/
    default: return '?';   /*desconocido, no deberia aparecer*/
  }
}

char * ConvierteModo (mode_t m, char *permisos)
{
  strcpy (permisos,"---------- ");

  permisos[0]=LetraTF(m);
  if (m&S_IRUSR) permisos[1]='r';  /*propietario*/
  if (m&S_IWUSR) permisos[2]='w';
  if (m&S_IXUSR) permisos[3]='x';
  if (m&S_IRGRP) permisos[4]='r';   /*grupo*/
  if (m&S_IWGRP) permisos[5]='w';
  if (m&S_IXGRP) permisos[6]='x';
  if (m&S_IROTH) permisos[7]='r';   /*resto*/
  if (m&S_IWOTH) permisos[8]='w';
  if (m&S_IXOTH) permisos[9]='x';
  if (m&S_ISUID) permisos[3]='s';  /*setuid, setgid y stickybit*/
  if (m&S_ISGID) permisos[6]='s';
  if (m&S_ISVTX) permisos[9]='t';
  return permisos;
}

char * ConvierteModo2 (mode_t m)
{
  static char permisos[12];
```

```
  strcpy (permisos,"---------- ");

  permisos[0]=LetraTF(m);
  if (m&S_IRUSR) permisos[1]='r';  /*propietario*/
  if (m&S_IWUSR) permisos[2]='w';
  if (m&S_IXUSR) permisos[3]='x';
  if (m&S_IRGRP) permisos[4]='r';   /*grupo*/
  if (m&S_IWGRP) permisos[5]='w';
  if (m&S_IXGRP) permisos[6]='x';
  if (m&S_IROTH) permisos[7]='r';   /*resto*/
  if (m&S_IWOTH) permisos[8]='w';
  if (m&S_IXOTH) permisos[9]='x';
  if (m&S_ISUID) permisos[3]='s';  /*setuid, setgid y stickybit*/
  if (m&S_ISGID) permisos[6]='s';
  if (m&S_ISVTX) permisos[9]='t';
  return (permisos);
}

char * ConvierteModo3 (mode_t m)
{
  char * permisos;
  permisos=(char *) malloc (12);
  strcpy (permisos,"---------- ");

  permisos[0]=LetraTF(m);
  if (m&S_IRUSR) permisos[1]='r';  /*propietario*/
  if (m&S_IWUSR) permisos[2]='w';
  if (m&S_IXUSR) permisos[3]='x';
  if (m&S_IRGRP) permisos[4]='r';   /*grupo*/
  if (m&S_IWGRP) permisos[5]='w';
  if (m&S_IXGRP) permisos[6]='x';
  if (m&S_IROTH) permisos[7]='r';   /*resto*/
  if (m&S_IWOTH) permisos[8]='w';
  if (m&S_IXOTH) permisos[9]='x';
  if (m&S_ISUID) permisos[3]='s';  /*setuid, setgid y stickybit*/
  if (m&S_ISGID) permisos[6]='s';
  if (m&S_ISVTX) permisos[9]='t';
  return (permisos);
```

```
}
```

Errors

When a system call cannot perform (for whatever reason) the task it was asked to do, it returns a special value (usually **-1**), and sets an external integer variable variable (**errno**) to an error code

The man page of the system call explains the reason why the system call produced such an error code.

A generic message explaning the error can be obtained with any of these methods:

- the *perror()* function prints that message to the standard error (the screen, if the standard error has not been redirected)
- the *strerror()* function returns the string with the error description if we supply it with the error code
- the external array of pointers, `extern char * sys_errlist[]`, contains the error descriptions indexed by error number so that `sys_errlist[errno]` has a description of the error associated with *errno*

WORK SUBMISSION

- Work must be done in pairs.
- Moodle will be used to submit the source code (you'll be informed of the exact procedure at a later time)
- The name of the main program will be `p1.c`, Program must be able to be compiled with `gcc p1.c`, **Optionally** a `Makefile` can be supplied so that all of the source code can be compiled with just `make`
- **ONLY ONE OF THE MEMBERS OF THE GROUP** will submit the source code. The names and logins of all the members of the group should be in the source code of the main programs (at the top of the file)
- Works submited not conforming to these rules will be disregarded.

DEADLINE: 23:00, Friday October the 22nd, 2021

ASSESSMENT: For each pair, it will be done in its corresponding group, during the lab classes