Terminus 1st Report

### Unai Fernandez, Haizea Garaño, Unai Salas, Aitor Malo and Mariusz Januszek

DESCRIPTION

The project consists on recreating Terminus game, designed in MIT. For that we will develop a shell that executes commands designed and implemented by us. Those commands are ls, cp, cd, less, pwd and touch. There’re still some commands to do, for example: move, greep and man.

For the game we need folders and files. We’ll create a script to create and give certain permissions to those files or folders. When the user advances in the game, the permissions will change and some files too.

These are the command we implemented for the project and the system calls and library functions used to make this commands work.

# **ls command**

# We used opendir library function to open the directory and the readdir library function to read the directory. To identify the directory where we are we used getenv library function. We’ll use getcwd system call in the future.

# Then all the files and directories are printed. We differentiate the files and the directories by using the stat system call. If the ***fstats.st\_mode & S\_IFDIR*** is true that mean that it is a directory otherwise it is a file.

NAME

ls - list directory contents

SYNOPSIS

ls [FILE]...

DESCRIPTION

List information about the FILEs (the current directory by default).

Sort entries alphabetically if none of -cftuvSUX nor --sort is speci‐

fied.

We differentiate the files and the directories by using the stat system call. If the ***fstats.st\_mode & S\_IFDIR*** is true that mean that it is a directory otherwise it is a file.

Mandatory arguments to long options are mandatory for short options

too.

SEE ALSO

open

REPORTING BUGS

# **cp command**

# For this command we used open system call to open the files as arguments, read system call to read all the information in file1 (argv[1]).

# We also used write to write the info of the first file (argv[1]) to the second one (argv[2]).

# NAME

# cp - copy files and directories

# SYNOPSIS

# cp [OPTION]... [-T] SOURCE DEST

# cp [OPTION]... SOURCE... DIRECTORY

# cp [OPTION]... -t DIRECTORY SOURCE...

# DESCRIPTION

# Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

SEE ALSO

-open()

-write()

REPORTING BUGS

# **cd command**

NAME

cd: This command changes the shell working directory and prints the name of the new path by the console.

SYNOPSIS

cd [dir].

DESCRIPTION

Change the current directory to “dir”. The value of the home shell variable is the default directory. We get the current working directory and we print it.

Then we change the directory to our input directory and we obtain the directory after the change, printing the name of this new path.

SEE ALSO

-bash(1)

-chdir()

-getcwd()

REPORTING BUGS

The chdir() system call is not being well executed because the programm does not change the directory.

# **less command**

For this command we have used fgets c library function, fputs, fopen and some system calls as open and read. With this command we have the ability to read on the screen everything that is within an item.

NAME

less - opposite of more

SYNOPSIS

**less** **[-[+]aABcCdeEfFgGiIJKLmMnNqQrRsSuUVwWX~]**

**[-b** space**]** **[-h** lines**]** **[-j** line**]** **[-k** keyfile**]**

**[-{oO}** logfile**]** **[-p** pattern**]** **[-P** prompt**]** **[-t** tag**]**

**[-T** tagsfile**]** **[-x** tab**,...]** **[-y** lines**]** **[-[z]** lines**]**

**[-#** shift**]** **[+[+]**cmd**]** **[--]** **[**filename**]...**

(See the OPTIONS section for alternate option syntax with long option

names.)

DESCRIPTION

Less is a program similar to more (1), but which allows backward move-

ment in the file as well as forward movement. Also, less does not have

to read the entire input file before starting, so with large input

files it starts up faster than text editors like vi (1). Less uses

termcap (or terminfo on some systems), so it can run on a variety of

terminals. There is even limited support for hardcopy terminals. (On

a hardcopy terminal, lines which should be printed at the top of the

screen are prefixed with a caret.)

Commands are based on both more and vi. Commands may be preceded by a

decimal number, called N in the descriptions below. The number is used

by some commands, as indicated.

**pwd command**

NAME

pwd - print name of current/working directory

SYNOPSIS

pwd

DESCRIPTION

Print the full filename of the current working directory.

**grep command**

NAME

grep, egrep, fgrep, zgrep, zegrep, zfgrep -- file pattern searcher

SYNOPSIS

**grep** [**-abcdDEFGHhIiJLlmnOopqRSsUVvwxZ**] [**-A** num] [**-B** num] [**-C**[num]]

[**-e** pattern] [**-f** file] [**--binary-files**=value] [**--color**[=when]]

[**--colour**[=when]] [**--context**[=num]] [**--label**] [**--line-buffered**]

[**--null**] [pattern] [file ...]

DESCRIPTION

The **grep** utility searches any given input files, selecting lines that

match one or more patterns. By default, a pattern matches an input line

if the regular expression (RE) in the pattern matches the input line

without its trailing newline. An empty expression matches every line.

Each input line that matches at least one of the patterns is written to

the standard output.

SEE ALSO

ed(1), ex(1), gzip(1), sed(1), re\_format(7)

REPORTING BUGS

# **mv command**

NAME

**mv** -- move files

SYNOPSIS

**mv** [**-f** | **-i** | **-n**] [**-v**] source target

**mv** [**-f** | **-i** | **-n**] [**-v**] source ... directory

DESCRIPTION

In its first form, the **mv** utility renames the file named by the source

operand to the destination path named by the target operand. This form

is assumed when the last operand does not name an already existing direc-

tory.

In its second form, **mv** moves each file named by a source operand to a

destination file in the existing directory named by the directory oper-

and. The destination path for each operand is the pathname produced by

the concatenation of the last operand, a slash, and the final pathname

component of the named file.

SEE ALSO

- cp(1)

- rm(1)

- symlink(7)

# **touch command**

NAME

touch - change file timestamps

SYNOPSIS

touch [OPTION]... FILE...

DESCRIPTION

Update the access and modification times of each FILE to the current time.

A FILE argument that does not exist is created empty, unless -c or -h is supplied.

A FILE argument string of - is handled specially and causes touch to change the times of the file associated with standard output.

SEE ALSO

-readdir()

-opendir()

REPORTING BUGS

# **shell**

We have used theoretical concepts used in class such as the use of the fork within the processes, or the system calls as read.