Operating Systems | Assignment-1: Readme

Krishnam Omar | 2020309 | CSAM

The shell is designed to take up the following commands:
Internal commands:

1. cd
2. echo
3. pwd

External commands:

1. ls
2. cat
3. date
4. rm
5. mkdir

Extra commands:

1. exit
2. help

For Using Makefile

For clearing all previous outputs and files use: `make clean`
For compiling all the C files present use: `make`
For running the shell use: `make run`

File Structure

`main_shel.c` is the main C file which needs to be run to start the shell. This file is responsible for taking all inputs, executing all other files and creating threads or forks.
`path.h` is header file made which helps all the files in the directory to access the global variable `directory_path`. This variable is responsible for storing current directory in which the shell is present.

`internal_cmd` is the folder which contains all the header files of all three internal commands. These commands do not require any fork or thread to execute, thus written in a `.h` file, so can be easily included in the `main_shell.c` file.
`external_cmd` is teh folder which contains all the c files of five external commands. `bin` is the folder which gets created upon execution of the makefile. This folder contains all the necessary binary files of all the external commands c files.
`extra_cmd` is a folder which contains one extra command header file that is `exit,` which is necessary for quitting the shell.

Commands

cd

This command helps to go inside or come back outside a directory.

Special Options given:

- `-P` uses the physical directory structure without following force symbolic links.
- `-L` resolve symboliclinks in DIR after processing instances of '..' (default)

Errors and corner cases handled:

- "no such file or directory", when a user attempts to `cd` into a nonexistent entry.
- "not a directory", when a user attempts to `cd` into anything that isn't a directory.
- "invalid option", when a user provides an invalid option.
- "too many arguments", when a user enters more than one entry.

echo

Printing multiple arguments given to it.

Special Options given:

- `-n` doesn't prints the trailing new line
- `-E` enables the interpretation of backslash escapes (default)

Errors and corner cases handled:

- Don't parse options after the first non-option token has been encountered.
- Don't handle invalid options.

pwd

Printing path of current working directory.

Special Options given:

- `-P` for avoiding symlinks in the print
- `-L` to resolve symbolic links (default)

Errors and corner cases handled:

- “invalid option”, when a user provides an invalid option.
- “too many arguments” is NOT handled. This is in accordance with how the Bash shell handles this.

ls

Printing all files and folder in the current directory.

Special Options given:

- `-a, —-all` for including files starting with .
- `-i, --inode` to print the index number too
- `-1` For printing one file per line.

Errors and corner cases handled:

- unrecognized option”, when a user provides an invalid option’s longform. ● “invalid option”, when a user provides an invalid option.
- “no such file or directory”, when a user attempts to `ls` a nonexistent entry.
- If multiple entries are specified, print the entry’s name first, and then its listing in the next line.
- If `all` is specified, first print the parent and current directory identifiers: ‘..’ and ‘.’, and then print the rest of the files in alphabetical order.
- If a regular file is specified, just print its name.

cat

Printing content of a given file

Special Options given:

- `-E, —-show-ends` display $ at end of each line
- `-n, --number` to print the line number too

Errors and corner cases handled:

- “unrecognized option”, when a user provides an invalid option’s longform.
- “invalid option”, when a user provides an invalid option.
- “is a directory”, when a user provides ‘cat’ with a directory name.
- “no such file or directory”, when a user attempts to `cat` a nonexistent entry.
- Handling ‘-’ input, or generally reading from stdin proved to be challenging since I wanted to exit the input-loop when the user forces an EOF through Ctrl+D.

Assumptions made:

- The user will not attempt pass ‘-’ as a file twice in one operation. Rationale: the EOF is passed into the subsequent stdin, and no input is taken.
- User does not pass a binary file.

date

Printing system date and time

Special Options given:

- `-u, —-utc, —-universal` print UTC time (default)
- `-R, --rfc-email` print in RFC 5322 format.

Errors and corner cases handled:

- “unrecognized option”, when a user provides an invalid option’s longform.
- “invalid option”, when a user provides an invalid option.
- “extra operand”, when a user provides any other argument other than the format string.
- “invalid date”, when the user does not provide a valid format specifier string (it should ideally start with a “+”).
- “multiple output formats specified”, when multiple output formats are specified (the format string coupled with the -R option).

rm

Removing files given in argument

Special Options given:

- `-f, —-force` ignores non-existant files
- `-v, --verbose` explains the process

Errors and corner cases handled:

- “unrecognized option”, when a user provides an invalid option’s longform.
- “invalid option”, when a user provides an invalid option.
- “missing operand”, when a user does not provide any pathname argument.
- “cannot create: Permission denied", when the user/program does not have sufficient permissions to create a new directory in the working directory.
- “cannot create: Is a directory", when the pathname is a directory.
- “cannot create: No such file or directory", when a user attempts to `rm` a nonexistent file.

mkdir

Creating a directory.

Special Options given:

- `-m, —-mode=MODE` set file mode (as in chmod)
- `-v, --verbose` explains the process

Errors and corner cases handled:

- “unrecognized option”, when a user provides an invalid option’s longform.
- “invalid option”, when a user provides an invalid option.
- “missing operand”, when a user does not provide any pathname argument.
- “invalid mode”, when an invalid mode is passed through -m/--mode option.
- “cannot create: Permission denied", when the user/program does not have sufficient permissions to create a new directory in the working directory.
- “cannot create: File exits", when a file or directory of the same name already exists.
- “cannot create: No such file or directory", when a component of the path prefix specified by path does not name an existing directory or path is an empty string.

exit

Exiting/Quitting the shell. Takes no argument from user.

help

Prints the menu printed at the start of the shell, listing down all available function.

Threads and Forks

By default all the external commands working on forking. By passing the argument `&t` after end of any external command will make it run on thread. This is handeled in the main function of `main_shell.c` . In forking, the external functions are called by `exec()` command but in threading, this is done by `system()` call.