Documentation

Prajna Vohra
2021345

About this shell

- This shell can handle 3 internal commands and 5 external commands
- To exit from the shell, we need to type ^C (ctrl C)
- For each of the external commands, there are 2 options- to call them using the fork() and exec() call method, or to call them using the thread method.
- To give commands for the method that uses pthread create(), the user needs to give a command such as this: “ls&t”, “cat&t” etc, ie- append &t to the first argument of the command.
- Normal commands if given would be executed using the fork() and exec() system calls.
- The internal commands are executed separately in the shell

Following are the command line options that I implemented in my shell for each command.

Internal Commands

cd [Cd stands for change directory. It allows the user to change the current working directory.]

Options handled:

- **cd[directoryname]**- allows the user to user to move into a subdirectory of the current working directory
- **cd .. -** allows the user to go a level up from the current directory

Error handling:

- If the user gives only one argument, ie “cd”, then the shell displays a “Wrong number of arguments error” and exits.
- If the chdir command to change the directory does not work, the shell throws an error message.

Test cases:

1) Current directory: home/praj/
   Input: cd lol
   pwd
   Output: home/praj/lol

2) Current directory: home/praj/lol
   Input: cd ..
   pwd
   Output: home/praj/

echo displays the text which is passed as an argument

Options handled:

- **echo “string”** displays the string on the shell console
- **echo -e “string having \n”** displays the string with newline wherever newline characters are found
- **echo -e “string having \c”** displays till the “\c" character is found, and stops displaying after that.

Error handling:

- If the user gives only one argument, ie “echo”, then the shell displays a “Wrong number of arguments error” and exits.

Test cases:

1) myshell>> echo hello world
   hello world

2) myshell>> echo -e hello\nworld
   hello
   world

3) myshell>> echo -e hello\cworld
   hello

4) myshell>> echo -e hello\nworld My name is\cprajna
   hello
   world My name is

Note: In this shell, echo -e command would only work for those strings that have either \n,\c,none or both of these characters.

pwd [displays the current working directory]

Options handled:

- pwd- prints the current directory
- pwd --help
- pwd --version

Test cases:

1) myshell>> pwd
   /home/praj/

2) myshell>> pwd --help
   pwd: pwd [-LP]
   Print the name of the current working directory.

   Options:

```
      -L        print the value of $PWD if it names the current working directory
      -P        print the physical directory, without any symbolic links

      By default, `pwd' behaves as if `-L' were specified.

      Exit Status:
      Returns 0 unless an invalid option is given or the current directory cannot be
read.
```

3) myshell>>pwd --version

```
pwd (GNU coreutils) 9.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Jim Meyering.
```

External Commands

ls [used to list directory content of files and directories]

Options handled:

- **ls** lists all the files in the current directory except for hidden files
- **ls -1** lists all the files in the current directory in separate lines
- **ls -a** lists all files including hidden files in the current directory

Error handling:

- In this shell, only 1 or 2 arguments can be given for ls. So, if the user gives more than 2 arguments, like“ls -1 -1”, then the shell displays a “Wrong number of arguments error” and exits.
- If the directory is not found, it gives an error message “This directory does not exist”
- If the directory is not readable, it gives an error message “Cannot read directory”
- If any other option/character is given with ls, for example “ls p”, it gives an error “Invalid option”

Note: The files are not displayed in any particular order

Test cases:

1) myshell>> ls

```
hello.c   bye.txt   directory1   newfile.py
```

2) myshell>> ls -1
```
hello.c
bye.txt
directory1
newfile.py
```

3) myshell>> ls -a
```
.   ..   .vscode   hello.c   bye.txt   directory1   newfile.py
```

cat[used to read content from a file and displays the content]

Options handled:

- **cat file1** reads data of file1 and displays it
- **cat -n file1** reads and displays data of file1 with line numbers
- **cat -e file1** reads and displays data of file1 with "$" symbol appended at the end of each line

Error handling:

- If the user gives only 1 argument, then the shell displays a "Wrong number of arguments error" and exits.
- If the filename given is not found, it throws an error message "File does not exist"
- If any other command is given, it throws an error message "Wrong command"

Note: The shell only handles cat command for a single file. Multiple files are not permitted

Test cases:

1) myshell>> cat hello.c
```
Hi my name

is prajna
```

2) myshell>> cat -n hello.c
```
1 Hi my name
2
3 is prajna
```

3) myshell>> cat -e hello.c
```
Hi my name$
$
is prajna$
```

rm[used to remove a file or a directory]

Options handled:

- **rm file1** deletes file1 if found
- **rm file1 file2 file3….** Deletes all the files given if found
- **rm -i file1 file2 …** confirms with the user to remove the files given
- **rm -d directory-** removes an empty directory

Error handling:

- If the user gives only 1 argument, then the shell displays a “Wrong number of arguments error” and exits.
- If the file to be removed given is not found, it throws an error message “File not found”
- In -d command, if the directory to be removed is not found, it throws an error message “Directory does not exist”
- The -d command gives an error when the given directory is not empty.

Note: rm can be used for multiple files as well. In the -i command, if the user types “y” when prompted with the message, the given file is removed, otherwise it is not. The -d command is only applicable for empty directories.

Test cases:

1)
```
myshell>> rm hello.c bye.txt
myshell>>ls
directory1   newfile.py
```

2)
```
myshell>> rm -i hello.c bye.txt
rm: remove regular file 'hello.c' ?y
rm: remove regular file 'bye.txt"?y
```

3)
```
myshell>> rm -d directory1
myshell>>ls
hello.c   bye.txt   newfile.py
```

date[used to get date and time]

Options handled:

- **date-** displays the current date and time in IST
- **date -u** displays the date and time in UTC
- **date -r filename** displays the date and time the file was last modified

Error handling:

- If the user gives an invalid command, an error message is thrown “Wrong command”
- In date -r, if the given file is not found, it throws an error message “File not found”

Note: date -r command would work only with a single file, it cannot work with multiple files.

Test cases:

4) myshell>> date
   Wed Oct 26 09:53:29 IST 2022

5) myshell>> date -u
   Wed Oct 26 04:23:55 UTC 2022

6) myshell>> rm -r file1
   Wed Oct 26 09:54:22 IST 2022

mkdir[used to create a new directory]

Options handled:

- **mkdir dir1** creates a new directory dir1
- **mkdir -v dir1 dir2 dir3-** creates multiple directories dir1,dir2,dir3 in the same current directory
- **mkdir -p dir1/dir2/dir3** creates directories inside directories

Error handling:

- If the user gives only 1 argument, then the shell displays a “Wrong number of arguments error” and exits.
- If directory could not be created (mkdir returns -1), then the shell throws an error message “Directory could not be created”
- If any other command is given, it shows “Wrong command”

Note: mkdir can be used to create multiple directories only with the -v command. If multiple directories are given in mkdir then it would throw an error

Test cases:

1) myshell>> mkdir d1
   myshell>>ls
   hello.c   bye.txt   directory1   newfile.py   d1

2) myshell>> mkdir -v d1 d2 d3
   myshell>>ls
   hello.c   bye.txt   directory1   newfile.py   d1   d2   d3

3) myshell>> mkdir -p d1/d2/d3
   myshell>>ls
   hello.c   bye.txt   newfile.py d1