CSE231-OS
Assignment1

Name:Pavit Singh
Roll no:2021178

Basic unix shell and commands implementation

To run this shell, I am running the command “make run” facilitated by my makefile. The makefile also has a clean function

```
run:
    gcc ls.c -o ls
    gcc cat.c -o cat
    gcc date.c -o date
    gcc mkdir.c -o mkdir
    gcc rm.c -o rm
    gcc hi.c
    clear
    ./a.out

clear:
    rm ls
    rm a.out
    rm cat
    rm date
    rm mkdir
    rm rm
```

Commands Implemented:

1)**echo**- It takes arguments and prints them out in the terminal. The format is echo[options][args].
Commands implemented:

- Normal echo command
- echo -n command - Prints arguments without printing next line.

Error cases: In case program doesn’t identify commands they are treated as normal arguments and are printed out.

```
./a.out
Unix$hell: echo test
test
Unix$hell: echo -n test
test Unix$hell: echo test with many words and sentences
test with many words and sentences
Unix$hell:
```

2)**pwd-** Prints out current directory user is in. The format is pwd[options]
Commands implemented:

- Normal pwd command
- pwd -l command implemented using getenv(), it shows symbolic position of current directory
- pwd -p command it shows current position of our working directory

Command implemented using getcwd library function.
Error cases: Invalid syntax is checked, also ensured that pwd -l and -p return different values.

```
Unix$hell: pwd
/Users/pait/Desktop/Coding/os/Assignment 1~ Unix$hell: cd
Unix$hell: pwd -l
/Users/pait/Desktop/Coding/os/Assignment 1
Unix$hell: pwd -p
/Users/pait
Unix$hell:
```

3)**cd-** Changes the directory the user is in. The format is cd[options][args]
Commands implemented:

- Normal cd command(Absolute and relative paths)
- cd ~ cd / cd .. commands for home, root and previous directory navigation respectively.

Command implemented using chdir library function.
Error cases: Cases of invalid commands or paths have been handled through error messages.

```
/Users/pait
Unix$hell: cd ..
Unix$hell: pwd
/Users~ Unix$hell: cd pait
Unix$hell: pwd
/Users/pait~ Unix$hell: cd Users/pait/Desktop
Unix$hell: pwd
/Users/pait/Desktop~ Unix$hell: cd /
Unix$hell: pwd
/~ Unix$hell:
```

4)**ls-** Used to display the content of the directory in which we currently are
Commands implemented:

- Normal ls command
- ls ~ ls / commands for home, root directory content listing respectively.
- ls sub directory command to list the contents of a sub directory without changing directory

Command implemented using dirent library, it was forked into new program using fork() and wait() commands, also implemented using threads.
Errors: Wrong commands and syntax handled through error messages. Eg. if Wrong subdirectory is listed.

```
Unix$hell: cd Desktop
Unix$hell: ls
.
..
.DS_Store
.localized
Coding
design stuff
Unix$hell: ls Coding
.
..
.DS_Store
python
A1_21178.java
Java
os
.vscode
c programming
Unix$hell: ls ~
.
..
.DS_Store
.localized
Coding
design stuff
Unix$hell:
```

```
Unix$hell: pwd
/Users/pait/Desktop~ Unix$hell: ls &t
Coding          design stuff
Unix$hell: ls Coding &t
A1_21178.java   Java            c programming   os              python
Unix$hell: ls / &t
Applications    Volumes         etc             sbin
Library         bin             home            tmp
System          cores           opt             usr
Users           dev             private         var
Unix$hell:
```

5)**date-** Used to display the content of the directory in which we currently are
Commands implemented:

- Normal date command
- date -u command for GMT date and time
- date -R command for RFC 2822 format

Command implemented using time.h library, it was forked into new program using fork() and wait() commands, also implemented using threads.
Errors: Wrong commands and syntax handled through error messages. Eg. if Wrong arguments entered

```
Unix$hell: date
Wed Oct 26 21:50:30 2022
Unix$hell: date -u
Wed Oct 26 16:20:33 2022
Unix$hell: date -R
Wed, 26 Oct 2022 21:50:36 +0530
Unix$hell: date &t
Wed 26 Oct 2022 21:50:52 IST
Unix$hell: date -u &t
Wed 26 Oct 2022 16:21:01 UTC
Unix$hell: date -R &t
Wed, 26 Oct 2022 21:51:16 +0530
Unix$hell:
```

6)**cat**- Used to display the content of a selected file
Commands implemented:

- Normal cat command
- Cat command for 2 files
- Cat -n command for Numbers being displayed
- Cat -e command for line demarcation using $

Command implemented using file handling and getline() function, it was forked into new program using fork() and wait() commands, also implemented using threads.

Errors: Wrong commands ,syntax and incorrect files handled through error messages. Eg. if incorrect files entered

```
Unix$hell: cat test.txt
test program
test executable
test process
Unix$hell: cat -n test.txt
1 test program
2 test executable
3 test process
Unix$hell: cat -e test.txt
test program $
test executable$
test process$
Unix$hell: cat test.txt test1.txt
test program
test executable
test process
test program
test executable
test process
Unix$hell:
```

7)**rm-** Used to remove a selected file
Commands implemented:

- Normal rm command
- Rm -v - (verbose) command tells us which file is being deleted.
- Rm -i command which asks for user confirmation.

Command implemented using remove() function, it was forked into new program using fork() and wait() commands, also implemented using threads.
Errors: Wrong commands ,syntax and incorrect files handled through error messages. Eg. if file removal was unsuccessful.

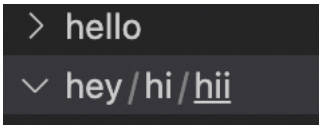
```
Unix$hell: rm ls
Unix$hell: rm -v test.txt
test.txt File successfully deletedUnix$hell: rm -i test1.txt
Are you sure you want to delete test1.txt file?(y/n)y
Unix$hell:
```

8)**mkdir-** Used to make a new directory
Commands implemented:

- Normal mkdir command
- mkdir -v - (verbose) command tells us which file is being deleted.
- mkdir -p command which can create recursive directories within a path

Command implemented using remove() function, it was forked into new program using fork() and wait() commands, also implemented using threads.
Errors: Wrong commands ,syntax and incorrect files handled through error messages. Eg. if file name already exists.

```
Unix$hell: mkdir -v hello
helloDirectory successfully createdUnix$hell: mkdir hello
Error in directory creationUnix$hell: mkdir -p hey/hi/hii
Unix$hell:
```



Additional Functionalities:

The shell is able to accommodate empty commands too.

```
Unix$hell:
Unix$hell:
Unix$hell:
Unix$hell:
Unix$hell:
Unix$hell:
Unix$hell:
```

An exit command is there too, to exit the program.

```
else if(strcmp(commandwords[0],"exit")==0){
    break;
}
```