NOTE : Before running the shell, please copy the folder named assign2 on your desktop and make one change in the code of shell according to your user name as follows:

Go to the main function of shell.c and see the third line in the main function –

```
char* baseAddress = "/home/akhil/Desktop/Assign2/";
```

Please replace the user name (akhil) above to the user name of your linux system.

Basic Description :

In the shell, first I take user input form the user and then I split that string input into tokens. Then, if the given command is an external command, then, first I create a child process (copy of parent parent process) using fork system call and in that child process I create a new process, which replaces the child process using the execv() system call and performs the desired work and then terminates (using exit system call). After that process terminates, then we come again back to the shell and then user can give input.

Commands, their options and their error handling :

1) ls : Prints the files and folders in working directory.

   Options :

   I. -F : Prints the files and fodders in working directory. The folder name will be ended by "/"
   II. --help : Prints information about this command.

   Errors handled :

   I. If any other option other than the above two options is provided, then it will give error message.
   II. If arguments (or options) greater than are passed, then it will give error message.
   III. If the there is error in the spelling of ls entered by user, then it will give error.
   IV. If the system call opendir() fails, due to some system failure, then it will give error message.

   Test cases:

   ```
   $ ls
   ```

   ```
   $ ls -F
   ```

   ```
   $ ls --help
   ```

2) cat:

   If you do not provide any argument and just press enter, then whatever text you will print, it will print exactly the same text again.

   If you specify the file names as arguments, then it will print the contents of the files. (You can provide more than one file at a time as argument).

Options:

a. -n: It will print the contents of the files by printing the line number also.
b. --help: Prints information about this command.

Errors handled:

I. If you just write the option -n and do not provide any file name as an argument, it will print error message/
II. If you specify a file name which does not exist, then it will print error message.

Test case:

```
$ cat

Hello                                    (You have to type this also or any text you like)

$ cat testFile.txt              (A file name testFile.txt is already present in the folder Assign2)

$ cat -n testFile.txt

$ cat --help
```

3) date: Prints the date and time in the default time zone of linux ubuntu system. (Generally, the default time zone of a linux system is PDT(Pacific Day Light time), so it prints according to that.)
Options:

I. -u:
It prints the date and time according to GMT time zone.
II. --help: Prints information about this command.

Errors handled;

a. If you provide an incorrect option, then it will give error.
b. If you provide more than the required arguments and option in the date command, then it will give error.

Test case :

```
$ date

$ date -u

$ date --help
```

4) mkdir: Creates directory.
   Options:
   a. -v: It prints the name of the directories, after creating it.
   b. --help: Prints information about this command.

   Errors handled:

   a. If you do not provide correct options, then it will give error.
   b. If you misspell the spell of mkdir, then it will give error.

Test case:

$ mkdir testDir2

$ mkdir -v testDir3

$ mkdir --help

5) rm: Removes the specified files.
   Options:
   a. -i: It asks a warning message that do you really want to delete the file. (Answer has to entered either y or n).
   b. --help: Prints information about this command.

   Errors handled:

   a. If you use -i option and in the warning message, you entered other that y or n, then it will give error and ask you again to enter.
   b. If you specify file name to be deleted, which does not exist, then it will give error.

   Test case:

   $ rm testFile2.txt (testFile2.txt has been provided)

   $ rm -i testFile3.txt (testFile3.txt has been provided)

   $ rm --help

6) cd: Changes the directory. (note that you can specify a directory whose name has spaces, then enquote in double quotes).

   Options:

   a. -L: If you use this option, then you can provide the symbolic path.
   b. --help: Prints information about this command.

   Errors handled:

a. If you specify a directory path which does not exist, then it will give error.
b. If you provide incorrect options, then it will give error.

Test case:

Please first create a directory, testDir.

```
$ cd testDir
```

```
$ cd -L testDir
```

```
$ cd --help
```

7) echo: Prints whatever you have written after it.
Options:
  a. -n: it print whatever you have supplied and then it prints the “terminal line tag” in the same line.
  b. --help: Prints information about this command.

  Errors/corner case handled:

  a. There is a corner case in which the user just provide the option -n but does not write after that what has to printed, then, then in that case, no error will be thrown and terminal tag line will be printed in the next line.
  b. If you don’ t provide the correct options, then it will give error.
  c. If you mis-spell echo, then it will give error.

Testcase:

```
$ echo hello
```

```
$ echo -n hello
```

```
$ echo --help
```

8) pwd: Prints the working directory.
Options:
1) -L : Prints the symbolic path.
2) --help: : Prints information about this command.

Errors:

a. If you provide incorrect options, then it will give error.
b. If you provide arguments or options more than the required, then it will give error.

Test case:

```
$ pwd
```

```
$ pwd -L
$ pwd --help
```

9) history: Prints the history of the commands. (Note that in the output console, that history command though which you just entered into the history console, in that console that recent history command will not be included.)

   Options:

   a. -a: It will append the commands from the history list to the history.txt file, where all commands are stored.
   b. --help: Prints information about this command.

   Errors:

   a. If you provide incorrect options, then it will give error.
   b. If you supply more arguments or options than required , it will print error.
   c. If you mis-spell history, then it will give error.

Testcase:

```
$ history

$ history -a

$ history --help
```

10) exit: Exits the shell.

   Options:

   a. --version : Prints the version information.
   b. --help: Prints information about this command.

   Errors:

a. If you provide incorrect options, then it will give error.
b. If you supply more arguments or options than required , it will print error.
c. If you mis-spell exit, then it will give error.

Test case :

```
$ exit

$ exit --version

$ exit -- help
```