LAB ASSIGNMENT 0 PART 1

➢ Internal commands:-

1. *Pwd:* It provides the current working directory, I have not offered any use of the flag, whether a flag is given or not it gives the same result.
   eg:- >>pwd

2. *History:* It provides the history of commands which was written in shell, whether it was successfully executed or not. It comes with a -c flag to clear all the previous history.
   eg: >> history

3. *Cd:* It helps in changing in the current working directory. I have assumed that the user provides the destination directory with a trailing "/" otherwise it throws an error that file doesn't exist and remain in the same shell. It also doesn't come with any flags. Also, that when entering only cd in shell, the user also presses an extra space before pressing the enter key.
   e.g.: >>cd /test

4. *Exit:* It helps in exiting the shell, and in the way, updating the history file.
   e.g. : >> exit

5. *Echo:* It prints out everything that comes after it. It comes with two flags viz -E and -n. "-n" helps in disabling the newline character from string and -E is the default flag of echo. It also handles the cases of multiple "/" (backslashes).
   e.g >>> echo Hello

➢ External commands:-

1. *Ls:* It prints all directories and files present in the current working directory
   e.g >>> ls -a

2. *Rm:* It helps in deleting a file or a directory, e.g >>> rm -r test

3. *Cat:* It helps in displaying the contents of the file in a different format with the use of flags. It just doesn't create any new file or append to any file, i.e., ">" operator doesn't work with this command.
   e.g. : >> cat -A test.txt

4. *mkdir:* It helps in creating directories. When trying to create a file whose name already exists, it throws an error.
   e.g. : >> mkdir test

5. *Date:* It shows the current date and time.
   e.g. : >> date -u

SANDEEP KUMAR SINGH
2018363
CSSS

- ✓ Each of the internal commands is handled by inbuilt functions of c, and each of the external commands is handled by their programs using fork(), execl() and wait() functions.

- ✓ Each of the external commands is handled by a different c file which uses “execv()” function to execute the command.

- ✓ There is a makefile which helps in compiling all files together and make it's binary named with “shell,” i.e., it can be executed with ./shell command. It also comes with the “clean” functionality where it deletes all the history of commands and the binary file created.

- ✓ My shell is also immune to multiple spaces between the arguments of commands, till there is at least one space between them.

- ✓ One assumption I have taken that user enters at least one space after every command.

SANDEEP KUMAR SINGH
2018363
CSSS