Exp 2 : Practice using shell

1. Objectives

1. To examine your search path and the value of the path variable.
2. To practice using shell.
3. To practice editing text files in pico, vi, and emacs

2. Lab Work

1. UNIX manual is divided into eight sections. Section 1 is for shell commands, section 2 is for system calls, and section 3 is for library calls (3C for C language library calls). Use the man command to get information about the following commands, system calls, and library calls: touch, cp, mv, rm, mkdir, rmdir, ls, lpr, cd, pwd, open, read, write, close, pipe, socket, mkfifo (command and C language library call), system, and printf. Complete the following table by adding a short description for each, including one or two typically used options for each command.

Command	Short Description	Example Use
touch		
cp		
mv		
rm		
mkdir		
rmdir		
ls		
lpr		
cd		
pwd		
open		
read		
write		
close		
pipe		
socket		
mkfifo		
system		
printf		

2. Display the message that is displayed when you log on. Hint: this message is stored in the /etc/motd file. Show your session.

3. Read through the following files, if they exist on your system: /etc/profile, ~/.profile, ~/.bashrc, ~/.bash_profile, ~/.login, and ~/.cshrc. What are the values of the following shell environment variables: PATH, path, LINES, HOME, and home. If you don't find any of these variables in the above files, use the echo $variable command to display the value of a variable, where variable may be PATH, path, LINES, HOME, pr home. Show your session.

4. To find the search path your system looks at or along to determine where the default shell finds programs, external commands or shell scripts, type echo $PATH (if you are using the Bourne, Korn or Bash shells) or type echo $path if you are using the C shell.

5. In UNIX, to execute a program, command or shell script, you type its name on the command line, possibly with options or arguments, and then press Enter. Given this procedure, if a program, command, or shell script is available on the system, but when you type its name on the command line, you get an error message saying that the program, command, or shell script does not exist, what do you think the problem is?

6. How did you know which shell you were using by default?

7. For your default shell, what is the name of the startup dot file? What is the PATH (path) variable defined as in this startup file?

8. Do you have the .profile file in your home directory, and what is the PATH (path) variable defined as in it? Is it the same path as from step b above?

9. Find out what shells are installed and available for your use on your UNIX system, and how to run them.

10. Use the appropriate command to run additional available shells on top of your default shell. What commands did you use to run the available shells? How can you know that the additional shells are actually running? Why would you want to run additional shells on top of your default login shell?

11. How can you terminate the shells that you started in exercise 10.above? What happens if you accidentally terminate the default login shell?

12. If you have started up two (2) additional shells on top of your default login shell, and want to keep running the default login shell and the 2nd shell you started up, how do you terminate the 1st shell you started up? (In other words, keep the default login shell and the third shell running, while terminating the 2nd shell.)How do you change the default prompt for the C or TC shell for the current session only? How do you change the default prompt for the Bourne or Korn shell for the current session only? To experiment with this option, depending on the shell you are running, change its default prompt to some new character for this login session only.

13. Change the default shell prompt for your login shell permanently. Capture your session here.

14. Change the default shell prompt for your login shell permanently. Capture your session here.

15. Explain the output of the following command(s), given the shell meta-characters included on the command line.

Command	Output
`ls ~ ; rm *.doc`	
`set prompt=`pwd``	
`man ps>pscoms`	
`ls .*.*`	
`lpr –Pspr [0-9]*.eps`	
`csh&`	
`mv "file 1 .ps" file1.ps`	
`echo `ls|wc –l` files`	
`echo 'space `" "` '`	

16. To get a preview of how shell scripts work, use your favorite text editor, and create the text shown below for the Bourne shell script file named read_demo. Then, execute the script file read_demo.

```
#!/bin/bash

# example of using arguments to a script
echo "My first name is $1"
echo "My surname is $2"
echo "Total number of arguments is $#"
```

17. execute the pico program on a new, blank file.

18. On the first line of the file, type your first and last name.

19. On the second line of the file, type 'The pico UNIX text editor allows you to do simple editing on small text files efficiently".

20. Use the <Ctrl-O> command to write the file to the default directory with the name lab5.

21. What version of pico did you use in the above work, and how did you find this out?

22. Use the cat command to create a short text file named shorty on your UNIX system, and then read that file into pico, and add text to it. What command did you use to read the cat-created file into pico?

23. Execute pico on your UNIX system using the –m command option. What functionality did the –m option give you in pico?