1. Working with ls

a) ls /usr/bin

The ls command displays the files present in the folder.

b) man ls

“man command” opens the manual entry of command

c) ls -l /usr/bin

The -l option shows the files in long listing format and the fifth column shows the size of each file.

d) ls -l /usr/bin

The first column in the -l option lists the permissions of the files

e) ls /usr/bin/a*

'*' is used to replace it with zero or more number of characters, so this command gives the files and directories starting with a.

. is used to denote the current directory and .. denotes the parent directory.

f) ls /usr/bin/*.pl

This will give the list of files of .pl type.

2. Creating and editing a file with emacs

a) emacs abc.txt

emacs creates the file and starts an editor.

b) Writing something in the file is simply done by typing something into it.

c) ctrl – x ctrl -s is used to save the file and ctrl -x ctrl -c is used to close the file.

3. Copying and removing a file

a) cp abc.txt abc1.txt

“cp source destination” copies the source file with the destination filename.

b) rm abc1.txt

"rm filename" removes the file with the name mentioned.

4. cat and redirecting output

   a) cat abc.txt

      "cat filename" displays the contents of the file on the screen,

   b) cat abc.txt > abc1.txt

      "cat file1 > file2" redirects the output of file1 to file2 instead of printing it to the screen.

   c) cat abc.txt >> abc1.txt

      "cat file1 >> file2" appends the output of file1 to file2, so the contents of file1 will come after the contents of file2.

   d) cat abc1.txt

      This will display the contents of abc1.txt where contents of abc.txt comes after abc1.txt

5. Looking at what is in a file

   a) head ex1.bed

      "head filename" will display the first 10 lines of the file.

      tail ex1.bed

      "tail filename" will display the last 10 lines of the file.

   b) more ex1.bed

      "more filename" displays the file one screenful at a time.

   c) head -n 5 ex1.bed

      "head -n k filename" , here the -n option prints the first k lines instead of first 10 lines.

      Tail -n 6 ex1.bed

      "tail -n k filename" : -n option prints k last lines instead of last 10 lines.

   d) less ex1.bed

less is similar to more, but it allows backward as well as forward movements.

6. Making and removing directories

a) mkdir test

"mkdir directory" creates a directory in the present path.

b) rmdir test

rmdir is used to delete an empty directory.

c) mkdir test1

d) cp abc.txt test1/

This will copy the file to the directory test1.

e) rmdir test1

This gives an error as test1 is not empty

f) rm -r test1

The -r option removes directories and their contents recursively.

g) mkdir 1 1/2 1/2/3 1/2/3/4

This makes the directory structure 1/2/3/4

7. Making the prompt prettier

a) echo $PS1

PS1 tells the command line how to display the prompt.

c) ls –a ~

-a option prints all the hidden directories

d) emacs ~/.bash_profile

The .bash_profile is executed for login shells.

e) The line means to display the prompt in different colors. To start coloring, this format is used \e[color\] and to end \e[m\] . \u means username, \h is hostname, \W is

current relative path, \$ is the prompt character. The last color sequence is not closed and so the remaining text will be in white color.

f) The username and hostname are in blue. The path and prompt charcter are in green. Rest of the line will be in white.

8. Count the number of characters

a) cd ~/class/ex1/

b) wc -m ex1.bd

-m option displays the number of characters in a file

c) wc -l ex1.bd

-l option displays the number of lines in a file.

9. Redirecting different streams

a) This command prints numbers from 1 to 100 and their halves on the screen as by default STDERR and STDOUT print their output on the screen.

b) perl -e 'foreach(1..100){print $_."\n"; print STDERR ($_ / 2)."\n"}' > out.txt

This redirects only the standard output to the file while the standard error is displayed on screen.

c) perl -e 'foreach(1..100){print $_."\n"; print STDERR ($_ / 2)."\n"}' 2> err.txt

The '2>' redirects only the standard error to the file.

d) perl -e 'foreach(1..100){print $_."\n"; print STDERR ($_ / 2)."\n"}' > out.txt 2>err.txt

e) perl -e 'foreach(1..100){print $_."\n"; print STDERR ($_ / 2)."\n"}' &> seq.txt

'&>' redirects both standard output and error to one file.

10. Piping data

a) The command prints numbers from 1 to 100 and their halves to the file seq.txt.

b) head seq.txt

   tail seq.txt

   more seq.txt

c) head -n 7 seq.txt

   tail -n 8 seq.txt

d) head -n 50 seq.txt | tail -n 1

e) tail -n 187 seq.txt

f) head -n 187 seq.txt

g) head -n 50 seq.txt | tail -n 6 | wc -m