Experiment No.: 3 **Date -06-03-2023**

Aim

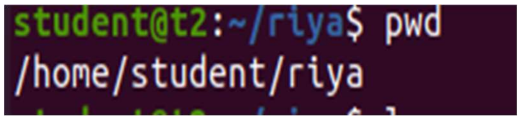
Familiarisation of Linux Commands

CO2

Perform System Aministration tasks

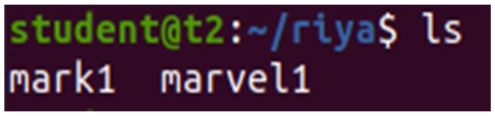
Procedure

$pwd



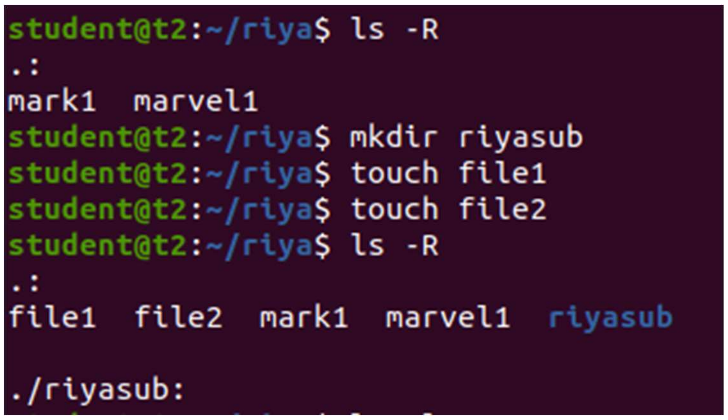
```
student@t2:~/riya$ pwd
/home/student/riya
```

$ls



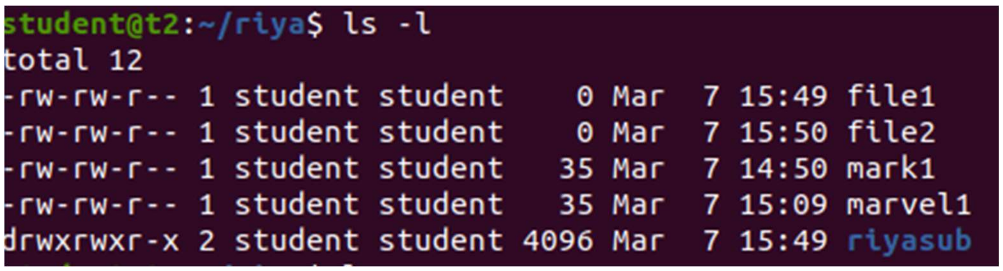
```
student@t2:~/riya$ ls
mark1  marvel1
```

$ls -R



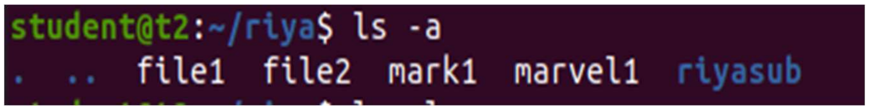
```
student@t2:~/riya$ ls -R
.:
mark1  marvel1
student@t2:~/riya$ mkdir riyasub
student@t2:~/riya$ touch file1
student@t2:~/riya$ touch file2
student@t2:~/riya$ ls -R
.:
file1  file2  mark1  marvel1  riyasub

./riyasub:
```

$ls -l



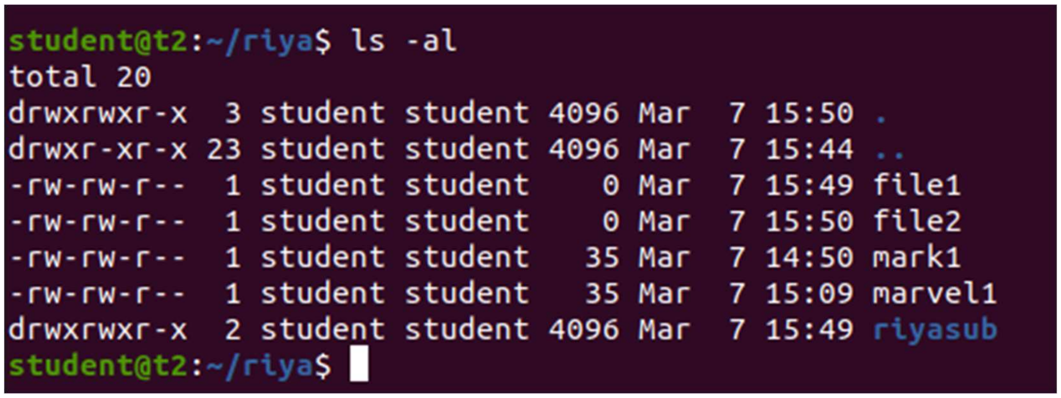
```
student@t2:~/riya$ ls -l
total 12
-rw-rw-r-- 1 student student    0 Mar  7 15:49 file1
-rw-rw-r-- 1 student student    0 Mar  7 15:50 file2
-rw-rw-r-- 1 student student   35 Mar  7 14:50 mark1
-rw-rw-r-- 1 student student   35 Mar  7 15:09 marvel1
drwxrwxr-x 2 student student 4096 Mar  7 15:49 riyasub
```

$ls -a



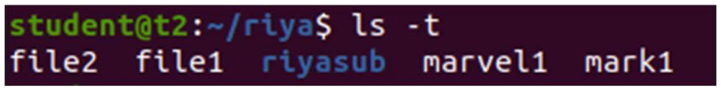
```
student@t2:~/riya$ ls -a
.  ..  file1  file2  mark1  marvel1  riyasub
```

$ls -al



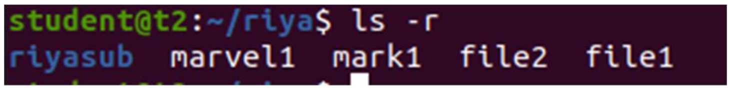
```
student@t2:~/riya$ ls -al
total 20
drwxrwxr-x  3 student student 4096 Mar  7 15:50 .
drwxr-xr-x 23 student student 4096 Mar  7 15:44 ..
-rw-rw-r--  1 student student    0 Mar  7 15:49 file1
-rw-rw-r--  1 student student    0 Mar  7 15:50 file2
-rw-rw-r--  1 student student   35 Mar  7 14:50 mark1
-rw-rw-r--  1 student student   35 Mar  7 15:09 marvel1
drwxrwxr-x  2 student student 4096 Mar  7 15:49 riyasub
student@t2:~/riya$ 
```

$ls -t



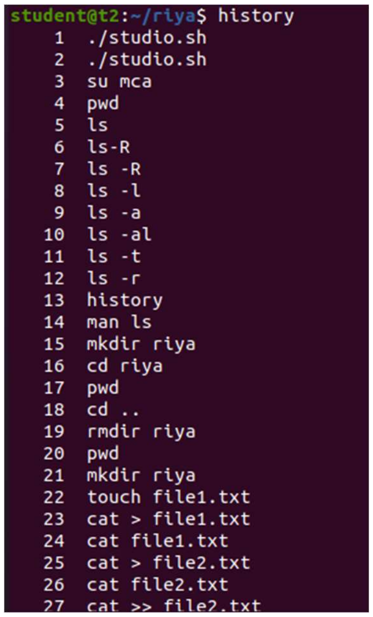
```
student@t2:~/riya$ ls -t
file2  file1  riyasub  marvel1  mark1
```

$ls -r



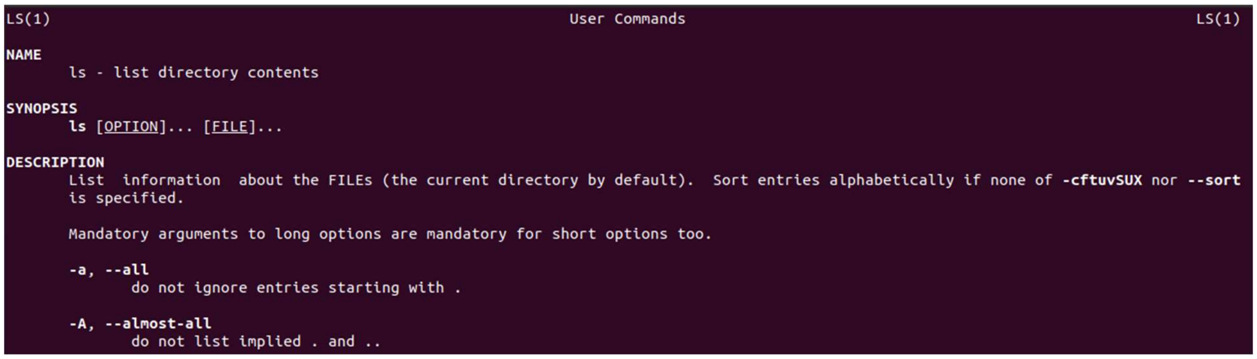
```
student@t2:~/riya$ ls -r
riyasub  marvel1  mark1  file2  file1
```

$history



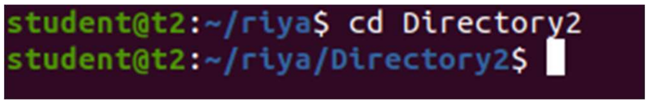
```
student@t2:~/riya$ history
    1  ./studio.sh
    2  ./studio.sh
    3  su mca
    4  pwd
    5  ls
    6  ls-R
    7  ls -R
    8  ls -l
    9  ls -a
   10  ls -al
   11  ls -t
   12  ls -r
   13  history
   14  man ls
   15  mkdir riya
   16  cd riya
   17  pwd
   18  cd ..
   19  rmdir riya
   20  pwd
   21  mkdir riya
   22  touch file1.txt
   23  cat > file1.txt
   24  cat file1.txt
   25  cat > file2.txt
   26  cat file2.txt
   27  cat >> file2.txt
```

$ man ls



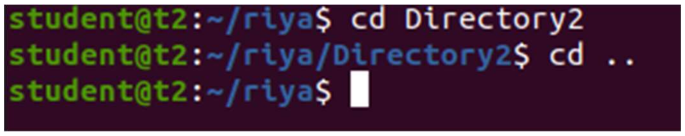
```
LS(1)                                   User Commands                                   LS(1)

NAME
       ls - list directory contents

SYNOPSIS
       ls [OPTION]... [FILE]...

DESCRIPTION
       List  information  about the FILEs (the current directory by default).  Sort entries alphabetically if none of -cftuvSUX nor --sort
       is specified.

       Mandatory arguments to long options are mandatory for short options too.

       -a, --all
              do not ignore entries starting with .

       -A, --almost-all
              do not list implied . and ..
```

$mkdir directoryname



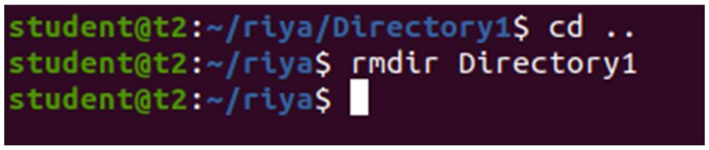
```
student@t2:~/riya$ cd Directory2
student@t2:~/riya/Directory2$
```

$ cd ..



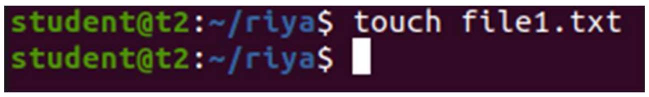
```
student@t2:~/riya$ cd Directory2
student@t2:~/riya/Directory2$ cd ..
student@t2:~/riya$
```

$ rmdir



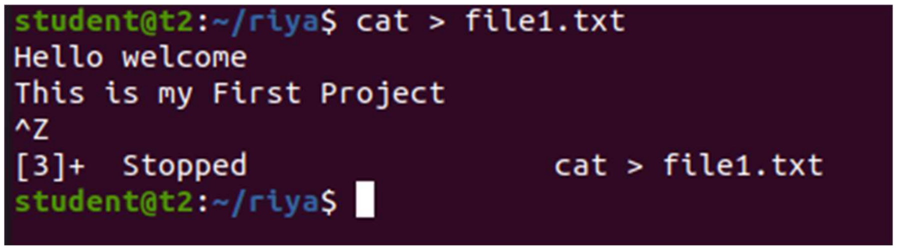
```
student@t2:~/riya/Directory1$ cd ..
student@t2:~/riya$ rmdir Directory1
student@t2:~/riya$
```

$touch filename



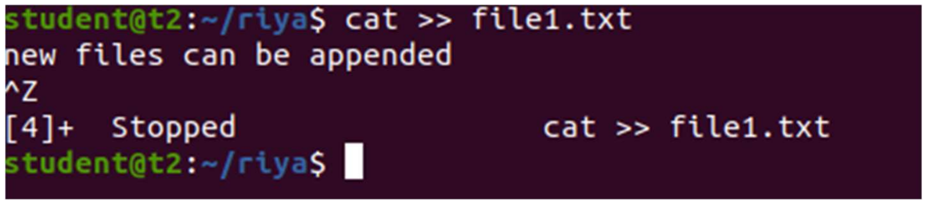
```
student@t2:~/riya$ touch file1.txt
student@t2:~/riya$ 
```

$cat > file1.txt



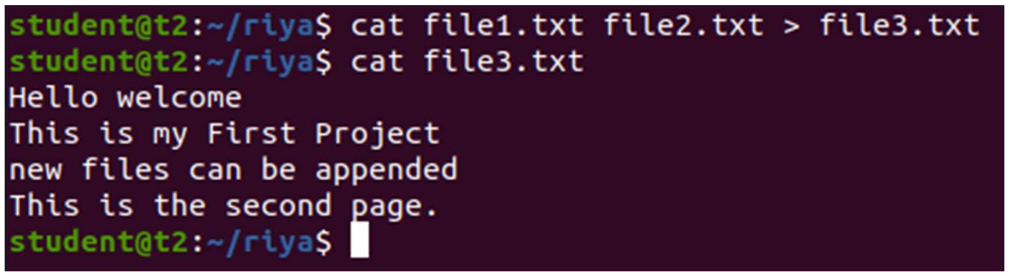
```
student@t2:~/riya$ cat > file1.txt
Hello welcome
This is my First Project
^Z
[3]+  Stopped                 cat > file1.txt
student@t2:~/riya$ 
```

$cat >> file1.txt



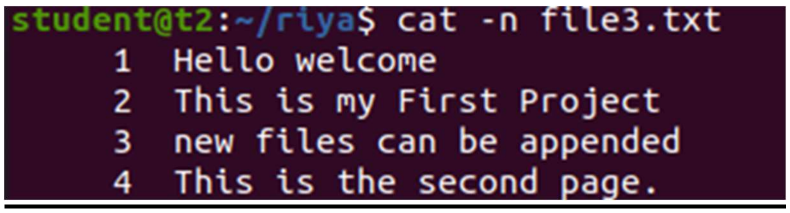
```
student@t2:~/riya$ cat >> file1.txt
new files can be appended
^Z
[4]+  Stopped                 cat >> file1.txt
student@t2:~/riya$ 
```

$cat file1.txt file2.txt > file3.txt



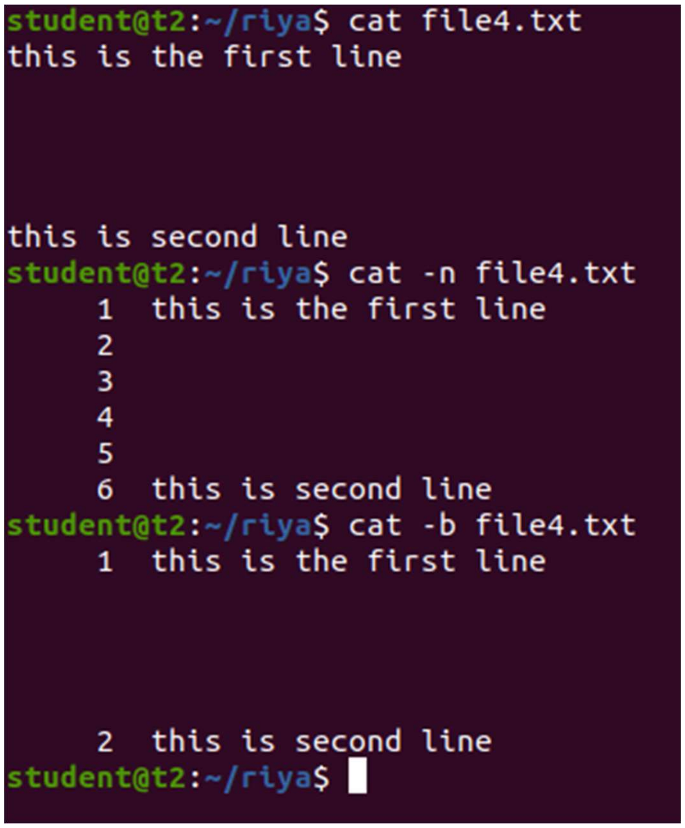
```
student@t2:~/riya$ cat file1.txt file2.txt > file3.txt
student@t2:~/riya$ cat file3.txt
Hello welcome
This is my First Project
new files can be appended
This is the second page.
student@t2:~/riya$ 
```

$cat -n file3.txt



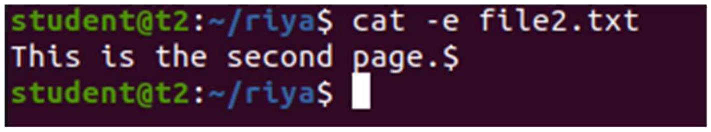
```
student@t2:~/riya$ cat -n file3.txt
     1  Hello welcome
     2  This is my First Project
     3  new files can be appended
     4  This is the second page.
```

$cat -b file4.txt



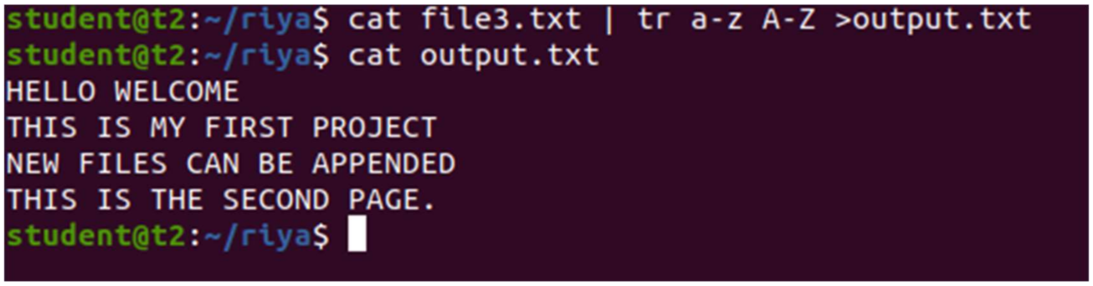
```
student@t2:~/riya$ cat file4.txt
this is the first line




this is second line
student@t2:~/riya$ cat -n file4.txt
     1  this is the first line
     2
     3
     4
     5
     6  this is second line
student@t2:~/riya$ cat -b file4.txt
     1  this is the first line




     2  this is second line
student@t2:~/riya$
```

$cat -e file2.txt



```
student@t2:~/riya$ cat -e file2.txt
This is the second page.$
student@t2:~/riya$
```

$cat c.txt | tr a-z A-Z > output.txt



```
student@t2:~/riya$ cat file3.txt | tr a-z A-Z >output.txt
student@t2:~/riya$ cat output.txt
HELLO WELCOME
THIS IS MY FIRST PROJECT
NEW FILES CAN BE APPENDED
THIS IS THE SECOND PAGE.
student@t2:~/riya$
```

<u>Result</u>

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

Experiment No.: 4 **Date -07-03-2023**

Aim

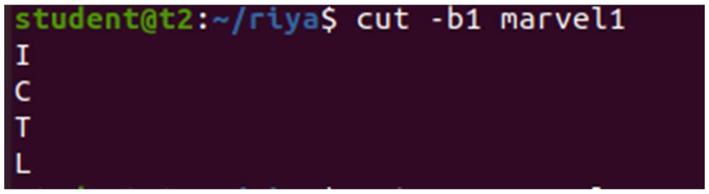
Familiarisation of Linux Commands

CO2

Perform System Aministration tasks

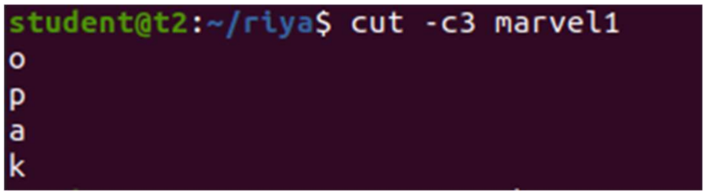
Procedure

$cut -b1 filename



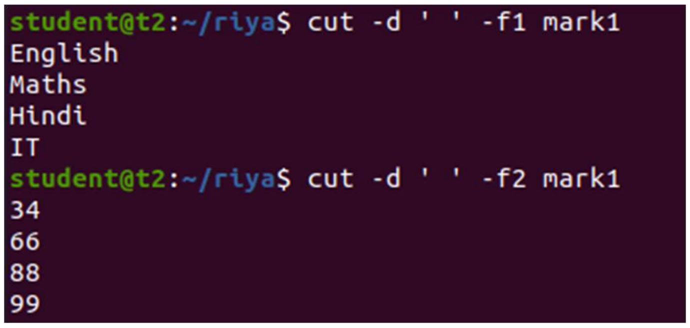
```
student@t2:~/riya$ cut -b1 marvel1
I
C
T
L
```

$cut -c3 filename



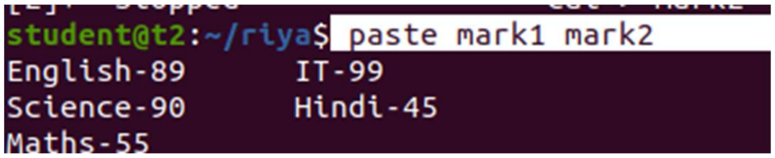
```
student@t2:~/riya$ cut -c3 marvel1
o
p
a
k
```

$cut -d ' ' -f1 filename



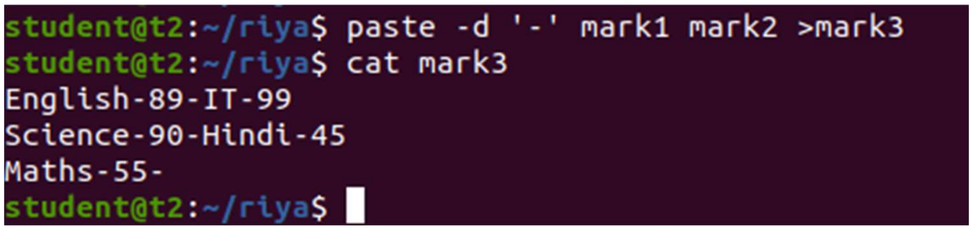
```
student@t2:~/riya$ cut -d ' ' -f1 mark1
English
Maths
Hindi
IT
student@t2:~/riya$ cut -d ' ' -f2 mark1
34
66
88
99
```

$paste mark1 mark2



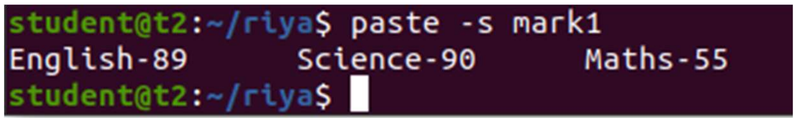
```
student@t2:~/riya$ paste mark1 mark2
English-89      IT-99
Science-90      Hindi-45
Maths-55
```

$paste -d ‘-’ mark1 mark2 > mark3



```
student@t2:~/riya$ paste -d '-' mark1 mark2 >mark3
student@t2:~/riya$ cat mark3
English-89-IT-99
Science-90-Hindi-45
Maths-55-
student@t2:~/riya$
```

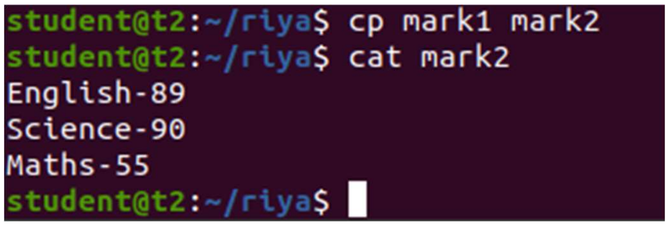
$paste -s mark1



```
student@t2:~/riya$ paste -s mark1
English-89      Science-90      Maths-55
student@t2:~/riya$
```

$cp mark1 mark2

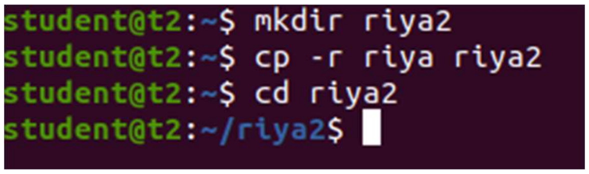
(content in mark1 is overwritten in mark2)



```
student@t2:~/riya$ cp mark1 mark2
student@t2:~/riya$ cat mark2
English-89
Science-90
Maths-55
student@t2:~/riya$
```

$cp -r riya riya2

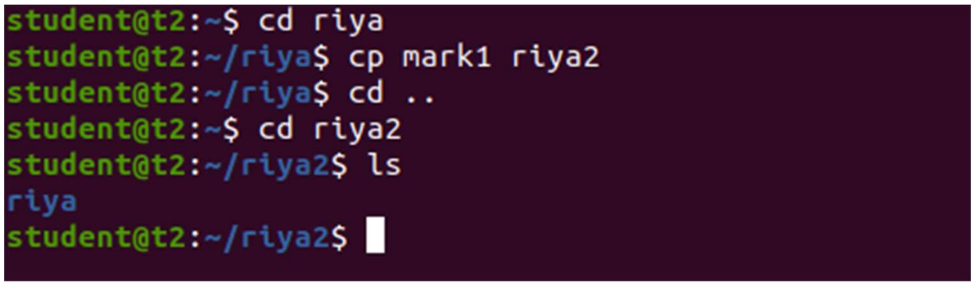
(to copy the directory along with its sub directories)



```
student@t2:~$ mkdir riya2
student@t2:~$ cp -r riya riya2
student@t2:~$ cd riya2
student@t2:~/riya2$
```

$cp filename directoryname

(to copy file from one directory to another directory)



```
student@t2:~$ cd riya
student@t2:~/riya$ cp mark1 riya2
student@t2:~/riya$ cd ..
student@t2:~$ cd riya2
student@t2:~/riya2$ ls
riya
student@t2:~/riya2$ 
```

Result

The program was executed and the result was successfully obtained. Thus CO2 was obtained.

Experiment No.: 5 **Date -13-03-2023**

Aim

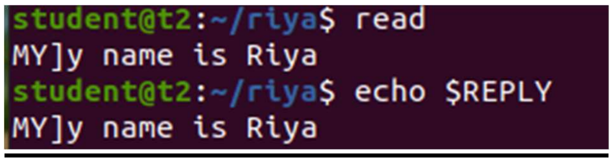
Familiarisation of Linux Commands

CO2

Perform System Aministration tasks

Procedure

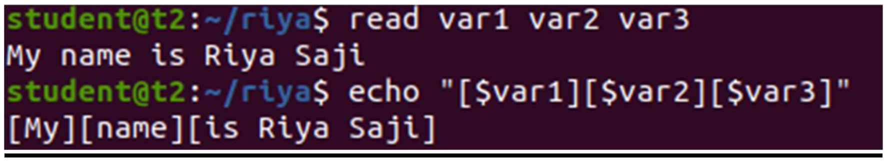
$read and echo



```
student@t2:~/riya$ read
MY]y name is Riya
student@t2:~/riya$ echo $REPLY
MY]y name is Riya
```

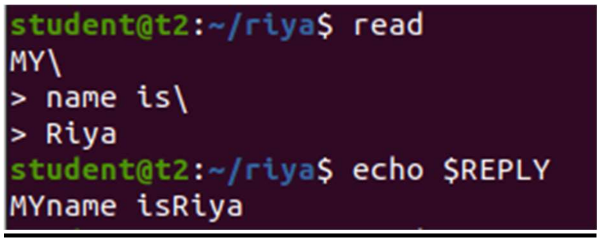
$read var1 var2 var3

(raeding the contents to 3 diffrent variables var1 var2 var3)



```
student@t2:~/riya$ read var1 var2 var3
My name is Riya Saji
student@t2:~/riya$ echo "[$var1][$var2][$var3]"
[My][name][is Riya Saji]
```

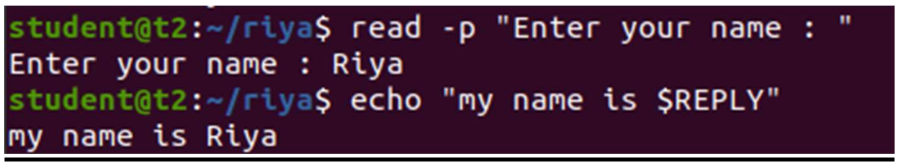
$read (the contents for multiple lines with \ at the end of each line)



```
student@t2:~/riya$ read
MY\
> name is\
> Riya
student@t2:~/riya$ echo $REPLY
MYname isRiya
```

$read -p

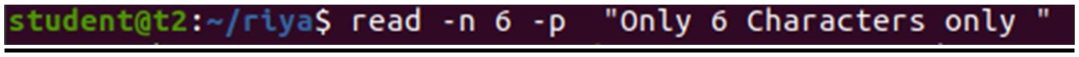
(for prompt text from user)



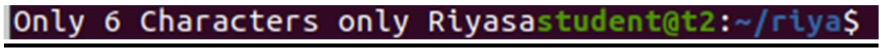
```
student@t2:~/riya$ read -p "Enter your name : "
Enter your name : Riya
student@t2:~/riya$ echo "my name is $REPLY"
my name is Riya
```

$read - n6

(specifying the limit for the content to read)



```
student@t2:~/riya$ read -n 6 -p  "Only 6 Characters only "
```

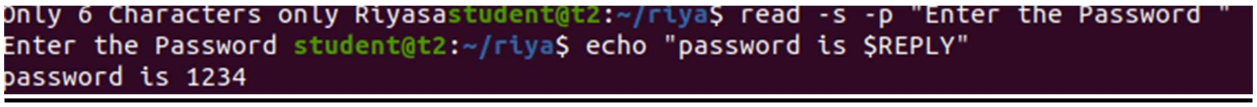


```
Only 6 Characters only Riyasastudent@t2:~/riya$
```

$read -s

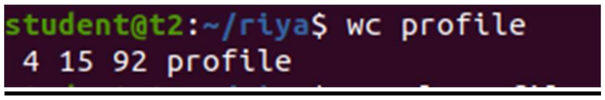
(security)

and $echo $REPLY (to display the password)



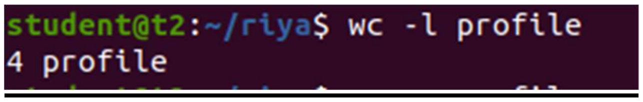
```
Only 6 Characters only Riyasastudent@t2:~/riya$ read -s -p "Enter the Password "
Enter the Password student@t2:~/riya$ echo "password is $REPLY"
password is 1234
```

$wc filename



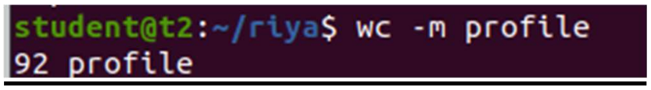
```
student@t2:~/riya$ wc profile
 4 15 92 profile
```

$wc -l filename



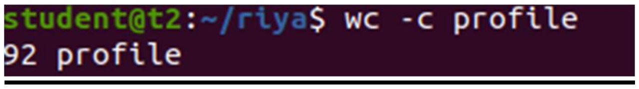
```
student@t2:~/riya$ wc -l profile
4 profile
```

$wc -m filename



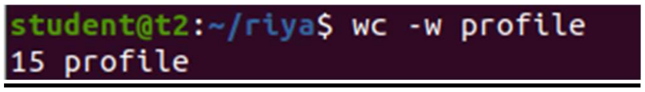
```
student@t2:~/riya$ wc -m profile
92 profile
```

$wc -c filename



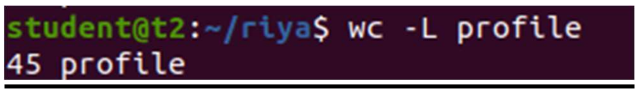
```
student@t2:~/riya$ wc -c profile
92 profile
```

$wc -w filename



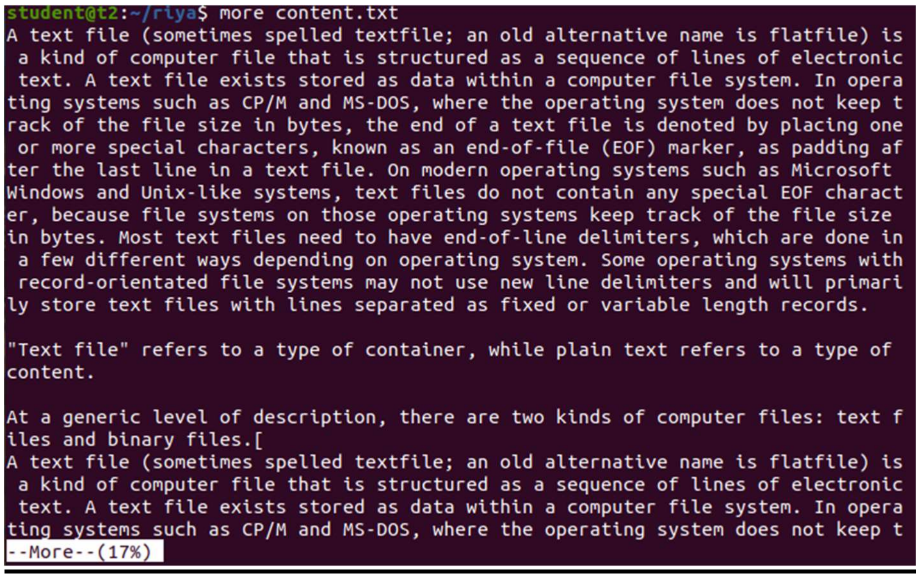
```
student@t2:~/riya$ wc -w profile
15 profile
```

$wc -L filename



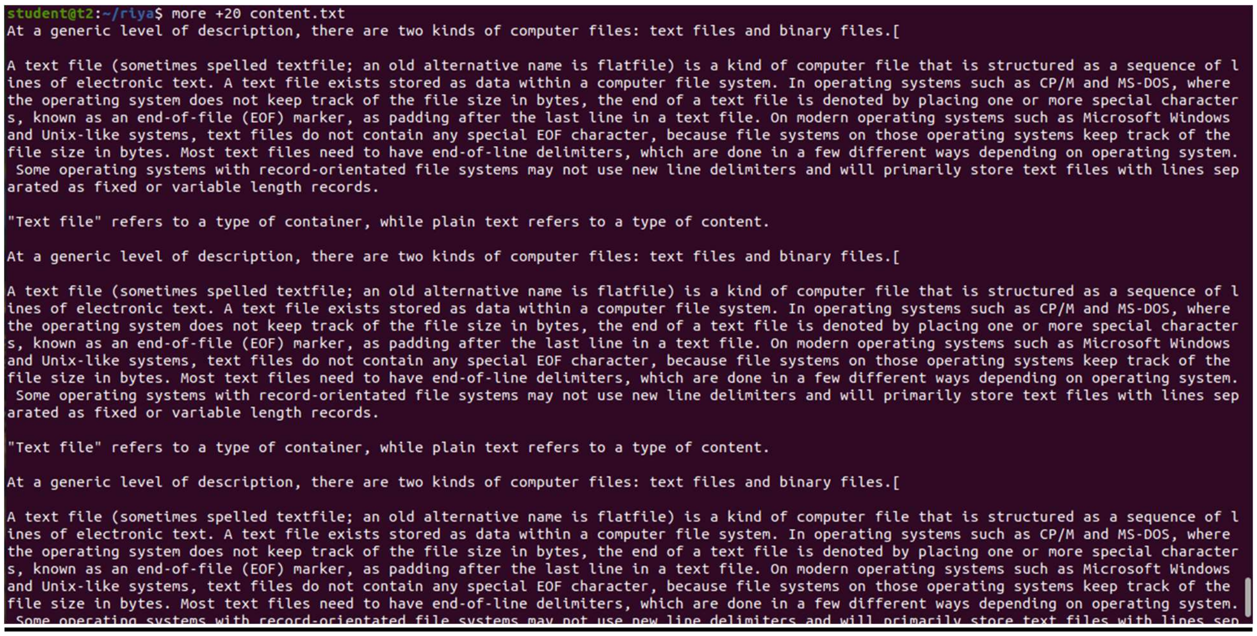
```
student@t2:~/riya$ wc -L profile
45 profile
```

$more content.txt



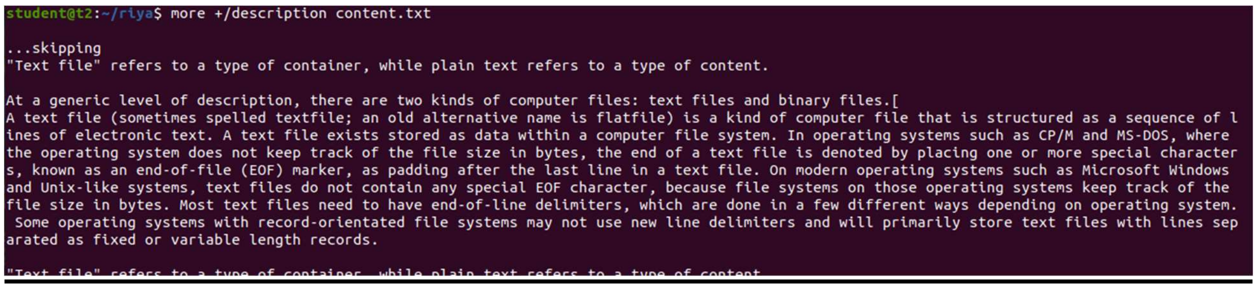
```
student@t2:~/riya$ more content.txt
A text file (sometimes spelled textfile; an old alternative name is flatfile) is
 a kind of computer file that is structured as a sequence of lines of electronic
 text. A text file exists stored as data within a computer file system. In opera
ting systems such as CP/M and MS-DOS, where the operating system does not keep t
rack of the file size in bytes, the end of a text file is denoted by placing one
 or more special characters, known as an end-of-file (EOF) marker, as padding af
ter the last line in a text file. On modern operating systems such as Microsoft
Windows and Unix-like systems, text files do not contain any special EOF charact
er, because file systems on those operating systems keep track of the file size
in bytes. Most text files need to have end-of-line delimiters, which are done in
 a few different ways depending on operating system. Some operating systems with
 record-orientated file systems may not use new line delimiters and will primari
ly store text files with lines separated as fixed or variable length records.

"Text file" refers to a type of container, while plain text refers to a type of
content.

At a generic level of description, there are two kinds of computer files: text f
iles and binary files.[
A text file (sometimes spelled textfile; an old alternative name is flatfile) is
 a kind of computer file that is structured as a sequence of lines of electronic
 text. A text file exists stored as data within a computer file system. In opera
ting systems such as CP/M and MS-DOS, where the operating system does not keep t
--More--(17%)
```

$more +20 content.txt



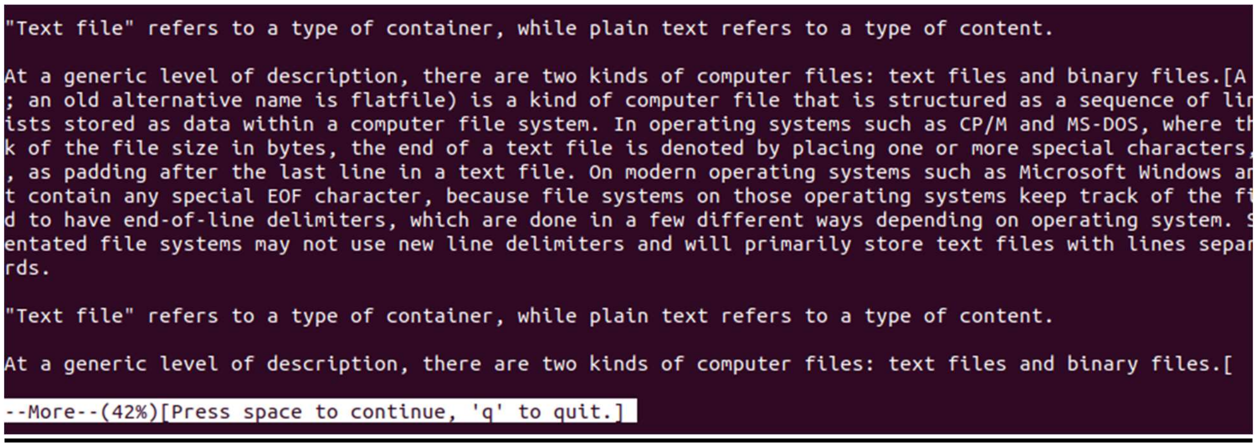
```
student@t2:~/riya$ more +20 content.txt
At a generic level of description, there are two kinds of computer files: text files and binary files.[

A text file (sometimes spelled textfile; an old alternative name is flatfile) is a kind of computer file that is structured as a sequence of l
ines of electronic text. A text file exists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where
the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special character
s, known as an end-of-file (EOF) marker, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows
and Unix-like systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the
file size in bytes. Most text files need to have end-of-line delimiters, which are done in a few different ways depending on operating system.
 Some operating systems with record-orientated file systems may not use new line delimiters and will primarily store text files with lines sep
arated as fixed or variable length records.

"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[

A text file (sometimes spelled textfile; an old alternative name is flatfile) is a kind of computer file that is structured as a sequence of l
ines of electronic text. A text file exists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where
the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special character
s, known as an end-of-file (EOF) marker, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows
and Unix-like systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the
file size in bytes. Most text files need to have end-of-line delimiters, which are done in a few different ways depending on operating system.
 Some operating systems with record-orientated file systems may not use new line delimiters and will primarily store text files with lines sep
arated as fixed or variable length records.

"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[

A text file (sometimes spelled textfile; an old alternative name is flatfile) is a kind of computer file that is structured as a sequence of l
ines of electronic text. A text file exists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where
the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special character
s, known as an end-of-file (EOF) marker, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows
and Unix-like systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the
file size in bytes. Most text files need to have end-of-line delimiters, which are done in a few different ways depending on operating system.
 Some operating systems with record-orientated file systems may not use new line delimiters and will primarily store text files with lines sep
```

$more +/pattern filename



```
student@t2:~/riya$ more +/description content.txt

...skipping
"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[
A text file (sometimes spelled textfile; an old alternative name is flatfile) is a kind of computer file that is structured as a sequence of l
ines of electronic text. A text file exists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where
the operating system does not keep track of the file size in bytes, the end of a text file is denoted by placing one or more special character
s, known as an end-of-file (EOF) marker, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows
and Unix-like systems, text files do not contain any special EOF character, because file systems on those operating systems keep track of the
file size in bytes. Most text files need to have end-of-line delimiters, which are done in a few different ways depending on operating system.
 Some operating systems with record-orientated file systems may not use new line delimiters and will primarily store text files with lines sep
arated as fixed or variable length records.

"Text file" refers to a type of container, while plain text refers to a type of content
```

$more -d filename



```
"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[A
; an old alternative name is flatfile) is a kind of computer file that is structured as a sequence of lin
ists stored as data within a computer file system. In operating systems such as CP/M and MS-DOS, where th
k of the file size in bytes, the end of a text file is denoted by placing one or more special characters,
, as padding after the last line in a text file. On modern operating systems such as Microsoft Windows an
t contain any special EOF character, because file systems on those operating systems keep track of the fi
d to have end-of-line delimiters, which are done in a few different ways depending on operating system. S
entated file systems may not use new line delimiters and will primarily store text files with lines separ
rds.

"Text file" refers to a type of container, while plain text refers to a type of content.

At a generic level of description, there are two kinds of computer files: text files and binary files.[

--More--(42%)[Press space to continue, 'q' to quit.]
```

Result

The program was executed and the result was successfully obtained. Thus CO2 was obtained.