Lab File

Operating System

(CSE 202)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Submitted to:	Submitted by:
Ms Seema Sharma	Shaina Mehta
Ast. Professor	A2305219268
CSE Department, ASET	B.tech. C.S.E.
	4CSE-4Y

AMITY SCHOOL OF ENGINEERING AND TECHNOLOGY

AMITY UNIVERSITY UTTAR PRADESH

NOIDA -201301

Exp No	Assignment Category	Code	Name of Experiment	Date of Allotment	Date of Evaluation	Max Marks	Marks Obtained	Faculty Sign
1			To explore the basic Linux commands.	17-12-2020	24-12-2020			
2			To explore file and directory related commands	24-12-2020	7-01-2021			
3			To explore advance Linux commands.	7-01-2021	14-01-2021			
4			To Explore More Advanced Linux Commands.	14-01-2021	21-01-2021			
5			Shell Scripting	21-01-2021	4-02-2021			
6			Shell Scripting	4-02-2021	11-02-2021			
7			Shell Scripting	11-02-2021	18-02-2021			
8			To simulate FCFS scheduling algorithm using C programming language.	18-02-2021	05-03-2021			

9			To simulate SJF scheduling algorithm using C programming language.	05-03-2021	18-03-2021			
10			To simulate Round Robin scheduling algorithm using C programming language.	24-03-2021	31-03-2021			
11			To simulate Banker's algorithm using C programming language.	24-03-2021	31-03-2021			
12			To simulate FIFO disk scheduling algorithm using C programming language.	24-03-2021	31-03-2021			
	Viva	Viva						

Experiment 1

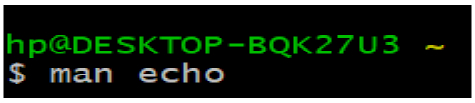
Date: 17-12-2020

Aim: To explore the basic Linux commands.

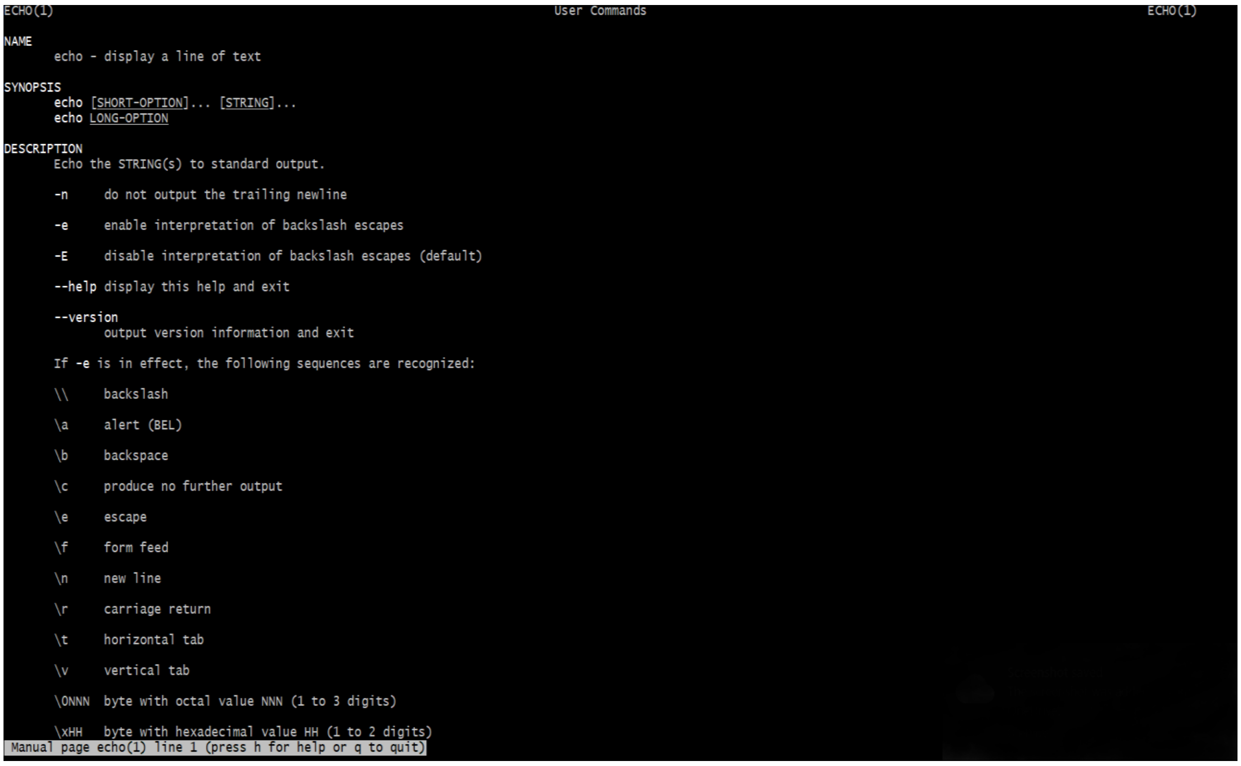
Software Used: Cgywin64 Terminal.

Theory:

1. **man**: man command provides the user with manual of other commands, type man with the name of the command.
   Syntax: man <command>

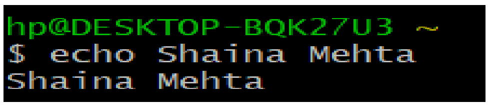


```
hp@DESKTOP-BQK27U3 ~
$ man echo
```



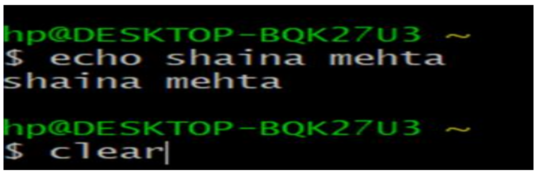
```
ECHO(1)                                User Commands                                ECHO(1)

NAME
       echo - display a line of text

SYNOPSIS
       echo [SHORT-OPTION]... [STRING]...
       echo LONG-OPTION

DESCRIPTION
       Echo the STRING(s) to standard output.

       -n     do not output the trailing newline

       -e     enable interpretation of backslash escapes

       -E     disable interpretation of backslash escapes (default)

       --help display this help and exit

       --version
              output version information and exit

       If -e is in effect, the following sequences are recognized:

       \\     backslash

       \a     alert (BEL)

       \b     backspace

       \c     produce no further output

       \e     escape

       \f     form feed

       \n     new line

       \r     carriage return

       \t     horizontal tab

       \v     vertical tab

       \0NNN  byte with octal value NNN (1 to 3 digits)

       \xHH   byte with hexadecimal value HH (1 to 2 digits)
 Manual page echo(1) line 1 (press h for help or q to quit)
```

2. **echo:** echo command is used to display the line of text/string that are passed as an argument.
   Syntax: echo <string to be displayed>

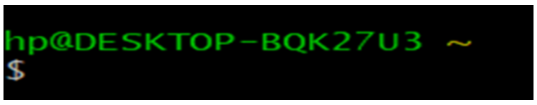


```
hp@DESKTOP-BQK27U3 ~
$ echo Shaina Mehta
Shaina Mehta
```

3. **clear:** clear command is used to clear the screen.
   Syntax: clear

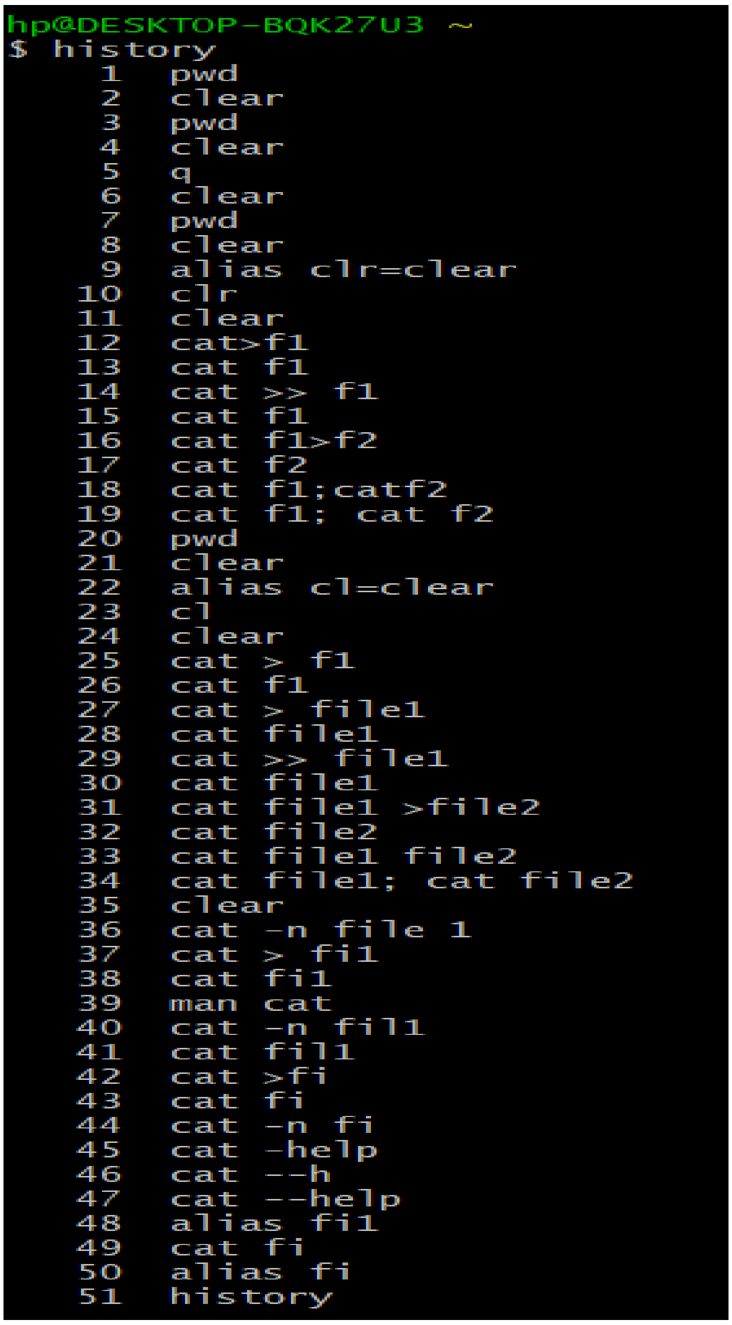


```
hp@DESKTOP-BQK27U3 ~
$ echo shaina mehta
shaina mehta

hp@DESKTOP-BQK27U3 ~
$ clear
```



```
hp@DESKTOP-BQK27U3 ~
$
```

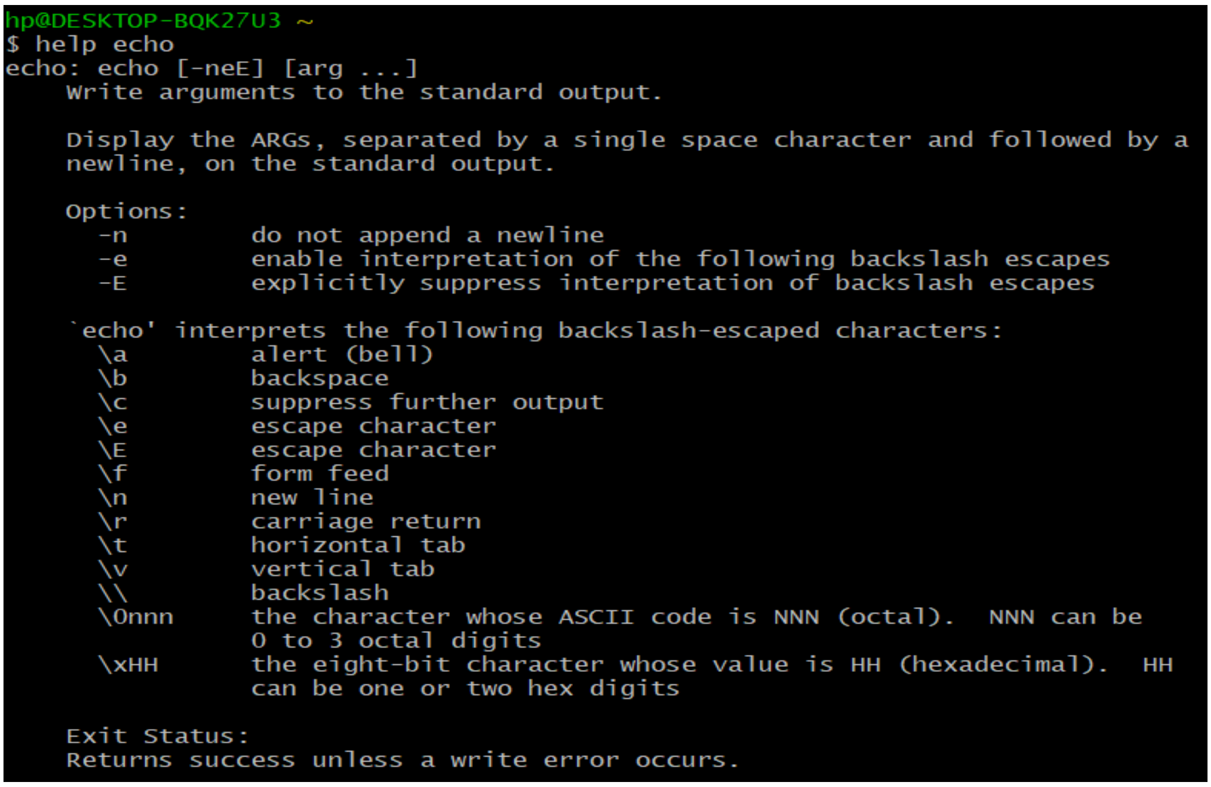
4. **history:** history command is used to view the commands one have entered before.
   Syntax: history



```
hp@DESKTOP-BQK27U3 ~
$ history
    1  pwd
    2  clear
    3  pwd
    4  clear
    5  q
    6  clear
    7  pwd
    8  clear
    9  alias clr=clear
   10  clr
   11  clear
   12  cat>f1
   13  cat f1
   14  cat >> f1
   15  cat f1
   16  cat f1>f2
   17  cat f2
   18  cat f1;catf2
   19  cat f1; cat f2
   20  pwd
   21  clear
   22  alias cl=clear
   23  cl
   24  clear
   25  cat > f1
   26  cat f1
   27  cat > file1
   28  cat file1
   29  cat >> file1
   30  cat file1
   31  cat file1 >file2
   32  cat file2
   33  cat file1 file2
   34  cat file1; cat file2
   35  clear
   36  cat -n file 1
   37  cat > fi1
   38  cat fi1
   39  man cat
   40  cat -n fi11
   41  cat fi11
   42  cat >fi
   43  cat fi
   44  cat -n fi
   45  cat -help
   46  cat --h
   47  cat --help
   48  alias fi1
   49  cat fi
   50  alias fi
   51  history
```

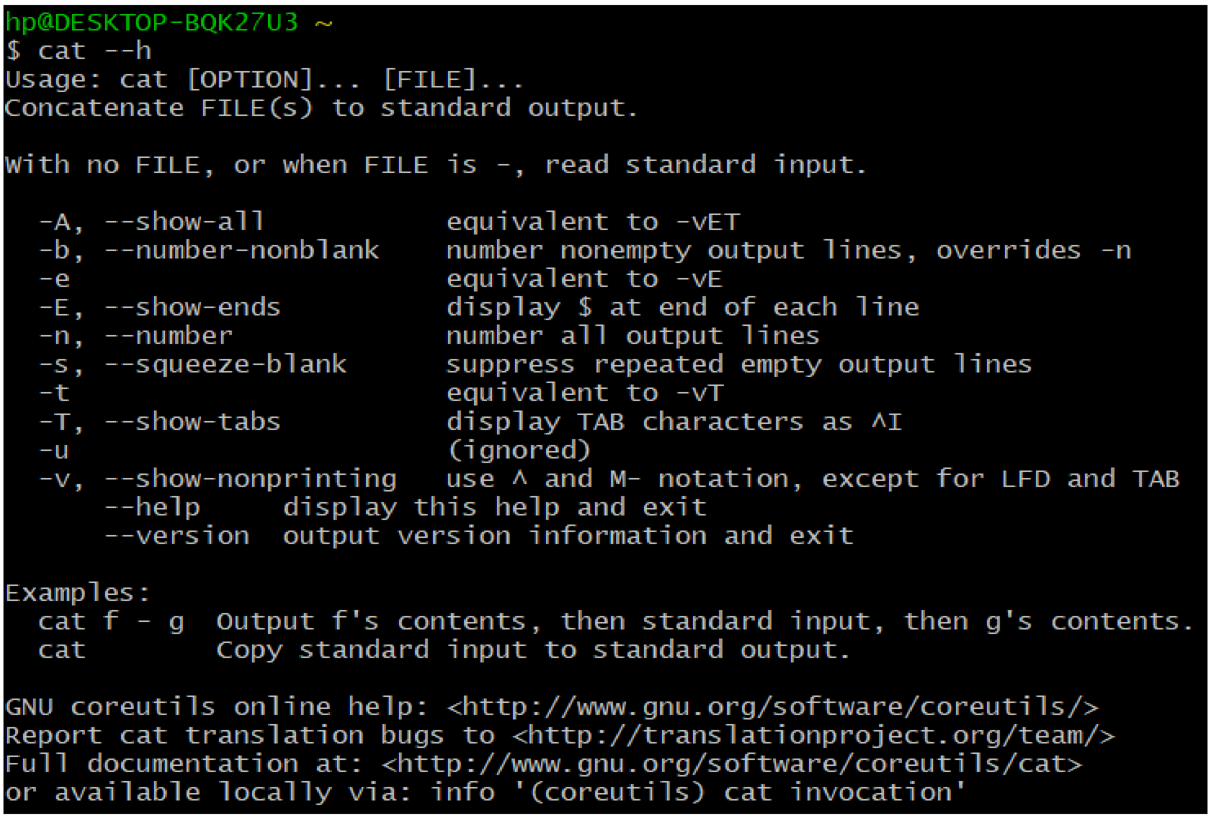
5. **help:** help command is used to display the information about shell build in commands.

Syntax: help <command> (used for echo command only) or <command> -- h (for all the commands except echo command) or <command> -- help (for all the commands except echo command)



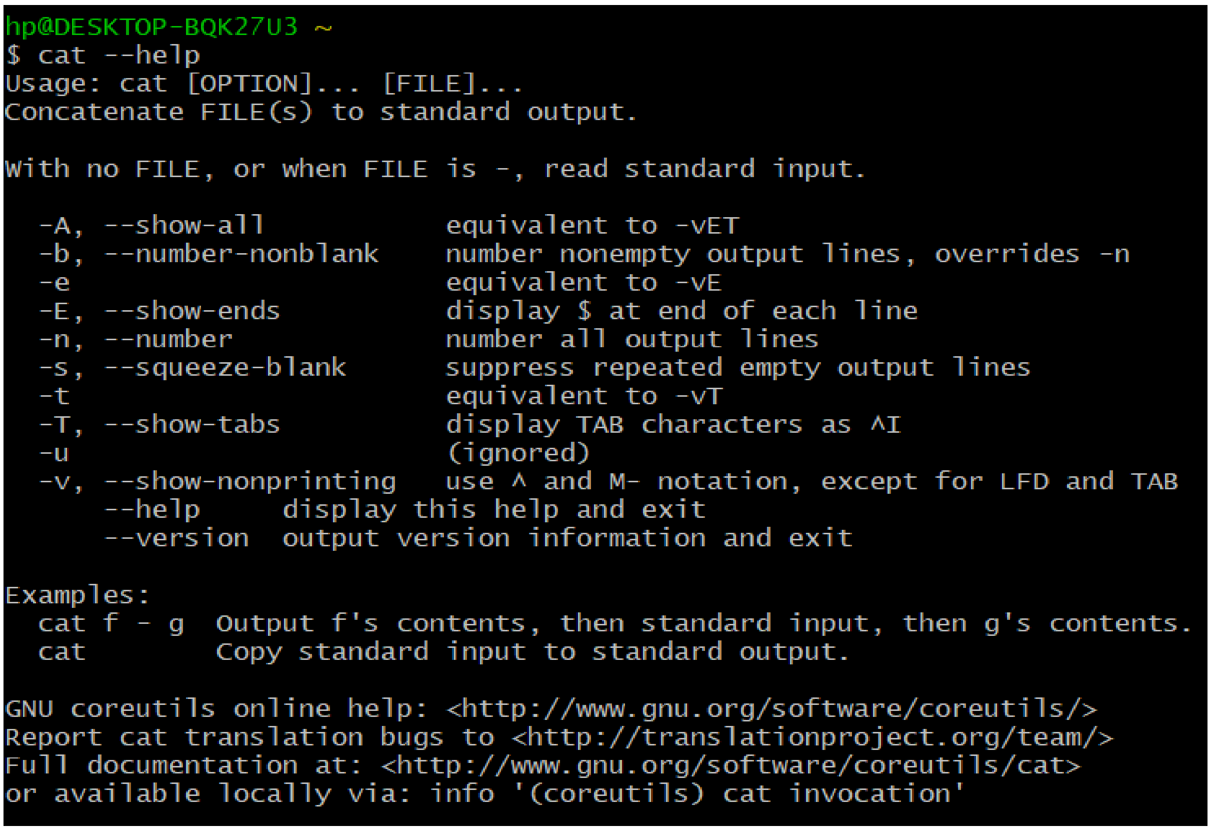
```
hp@DESKTOP-BQK27U3 ~
$ help echo
echo: echo [-neE] [arg ...]
    Write arguments to the standard output.

    Display the ARGs, separated by a single space character and followed by a
    newline, on the standard output.

    Options:
      -n        do not append a newline
      -e        enable interpretation of the following backslash escapes
      -E        explicitly suppress interpretation of backslash escapes

    `echo' interprets the following backslash-escaped characters:
      \a        alert (bell)
      \b        backspace
      \c        suppress further output
      \e        escape character
      \E        escape character
      \f        form feed
      \n        new line
      \r        carriage return
      \t        horizontal tab
      \v        vertical tab
      \\        backslash
      \0nnn     the character whose ASCII code is NNN (octal).  NNN can be
                0 to 3 octal digits
      \xHH      the eight-bit character whose value is HH (hexadecimal).  HH
                can be one or two hex digits

    Exit Status:
    Returns success unless a write error occurs.
```

Or



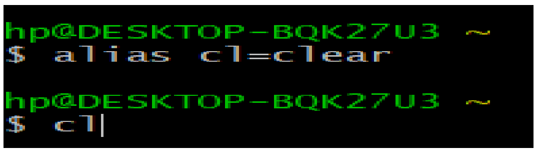
```
hp@DESKTOP-BQK27U3 ~
$ cat --h
Usage: cat [OPTION]... [FILE]...
Concatenate FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonempty output lines, overrides -n
  -e                       equivalent to -vE
  -E, --show-ends          display $ at end of each line
  -n, --number             number all output lines
  -s, --squeeze-blank      suppress repeated empty output lines
  -t                       equivalent to -vT
  -T, --show-tabs          display TAB characters as ^I
  -u                       (ignored)
  -v, --show-nonprinting   use ^ and M- notation, except for LFD and TAB
      --help     display this help and exit
      --version  output version information and exit

Examples:
  cat f - g  Output f's contents, then standard input, then g's contents.
  cat        Copy standard input to standard output.

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Report cat translation bugs to <http://translationproject.org/team/>
Full documentation at: <http://www.gnu.org/software/coreutils/cat>
or available locally via: info '(coreutils) cat invocation'
```

Or

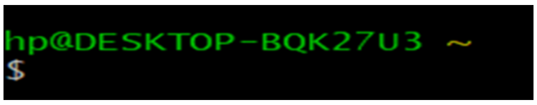


```
hp@DESKTOP-BQK27U3 ~
$ cat --help
Usage: cat [OPTION]... [FILE]...
Concatenate FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

  -A, --show-all           equivalent to -vET
  -b, --number-nonblank    number nonempty output lines, overrides -n
  -e                       equivalent to -vE
  -E, --show-ends          display $ at end of each line
  -n, --number             number all output lines
  -s, --squeeze-blank      suppress repeated empty output lines
  -t                       equivalent to -vT
  -T, --show-tabs          display TAB characters as ^I
  -u                       (ignored)
  -v, --show-nonprinting   use ^ and M- notation, except for LFD and TAB
      --help     display this help and exit
      --version  output version information and exit

Examples:
  cat f - g  Output f's contents, then standard input, then g's contents.
  cat        Copy standard input to standard output.

GNU coreutils online help: <http://www.gnu.org/software/coreutils/>
Report cat translation bugs to <http://translationproject.org/team/>
Full documentation at: <http://www.gnu.org/software/coreutils/cat>
or available locally via: info '(coreutils) cat invocation'
```

6. **alias:** alias command is used to create custom shortcuts to represent commands.
   Syntax: alias <alias name> = <command>

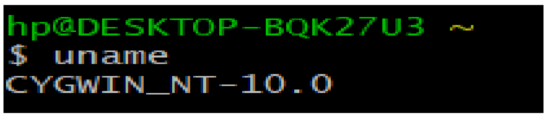


```
hp@DESKTOP-BQK27U3 ~
$ alias cl=clear

hp@DESKTOP-BQK27U3 ~
$ cl
```



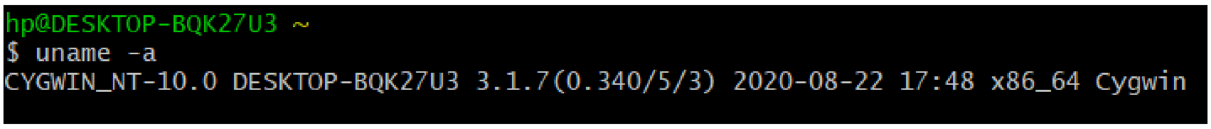
```
hp@DESKTOP-BQK27U3 ~
$
```

7. **uname:** uname command is used to print the basic information about your operating system (basically of Linux system) like machine name operating system kernel etcetera.
   Syntax: uname <options> or uname



```
hp@DESKTOP-BQK27U3 ~
$ uname
CYGWIN_NT-10.0
```

Or

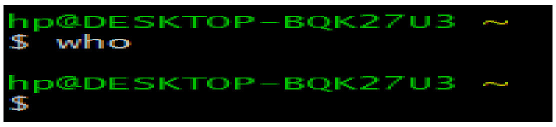


```
hp@DESKTOP-BQK27U3 ~
$ uname -a
CYGWIN_NT-10.0 DESKTOP-BQK27U3 3.1.7(0.340/5/3) 2020-08-22 17:48 x86_64 Cygwin
```

Note: Options used in uname command are:

- **-a, --all:** print all the system information, in the manner given above, except omit -p and -i if unknown.
- **-s, --kernel-name:** print the kernel name.
- **-n, --nodename:** print the network node hostname.
- **-r, --kernel-release:** print the kernel release.
- **-v, --kernel-version:** print the kernel version.
- **-m, --machine:** print the machine hardware name.
- **-p, --processor:** print the processor type (non-portable).
- **-i, --hardware-platform:** print the hardware platform (non-portable).
- **-o, --operating system:** print the operating system.
- **--help:** display this help and exit.
- **-version:** output version information and exit.

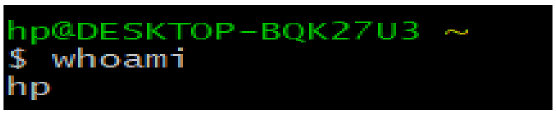
8. **who:** who command is used to give information of the currently logged in user on to the system. It displays login name of the users, terminal number and login time of the users.
   Syntax: who



```
hp@DESKTOP-BQK27U3 ~
$ who

hp@DESKTOP-BQK27U3 ~
$
```

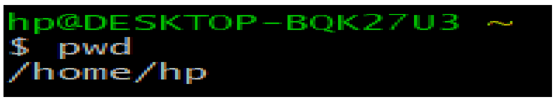
   Note: The information of the currently logged in user will not be display if we use emulator.

9. **whoami:** whoami command displays the username of the current user when this command is invoked. It is equivalent to id-un command.
   Syntax: whoami



```
hp@DESKTOP-BQK27U3 ~
$ whoami
hp
```

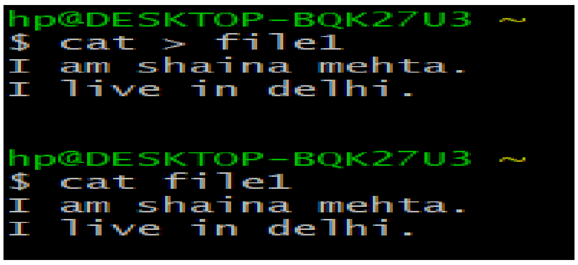
10. **pwd:** pwd command is used to display the current working directory.
    Syntax: pwd



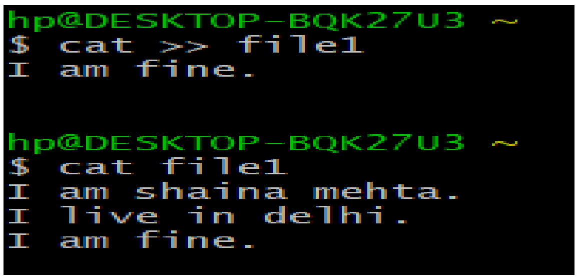
```
hp@DESKTOP-BQK27U3 ~
$ pwd
/home/hp
```

11. **cat:** cat command is used to create single or multiple files, view content of file/s, concatenate files and redirect output in terminal or files.
    Syntax:
    - cat > filename - to create a new file
    - cat filename - to open a file
    - cat >> filename - to append the content of a file
    - cat file1>file2 - to copy content of file 1 into file 2

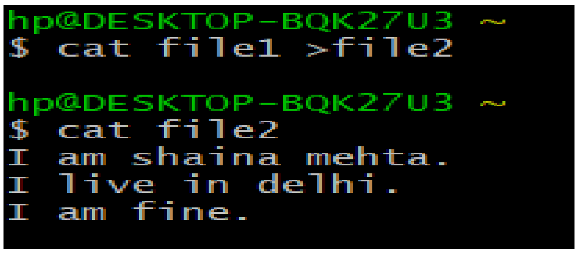
- cat file1; cat file2 – to open two files simultaneously which can be achieved by using semicolon which is used to perform multiple operations at the same time.
- Cat file1 file2 – to pen two files simultaneously.



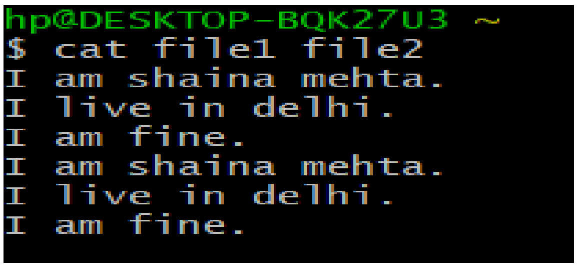
```
hp@DESKTOP-BQK27U3 ~
$ cat > file1
I am shaina mehta.
I live in delhi.

hp@DESKTOP-BQK27U3 ~
$ cat file1
I am shaina mehta.
I live in delhi.
```



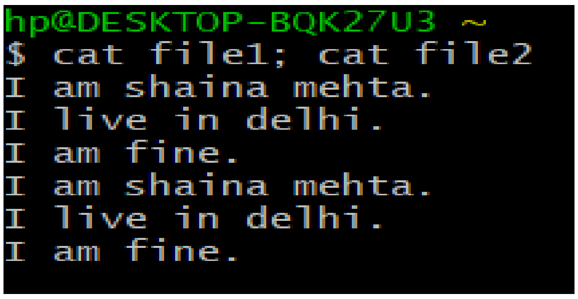
```
hp@DESKTOP-BQK27U3 ~
$ cat >> file1
I am fine.

hp@DESKTOP-BQK27U3 ~
$ cat file1
I am shaina mehta.
I live in delhi.
I am fine.
```



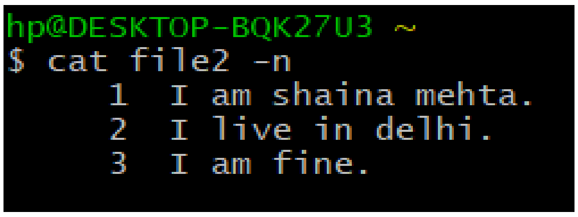
```
hp@DESKTOP-BQK27U3 ~
$ cat file1 >file2

hp@DESKTOP-BQK27U3 ~
$ cat file2
I am shaina mehta.
I live in delhi.
I am fine.
```



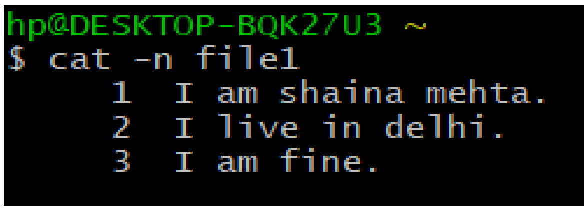
```
hp@DESKTOP-BQK27U3 ~
$ cat file1 file2
I am shaina mehta.
I live in delhi.
I am fine.
I am shaina mehta.
I live in delhi.
I am fine.
```



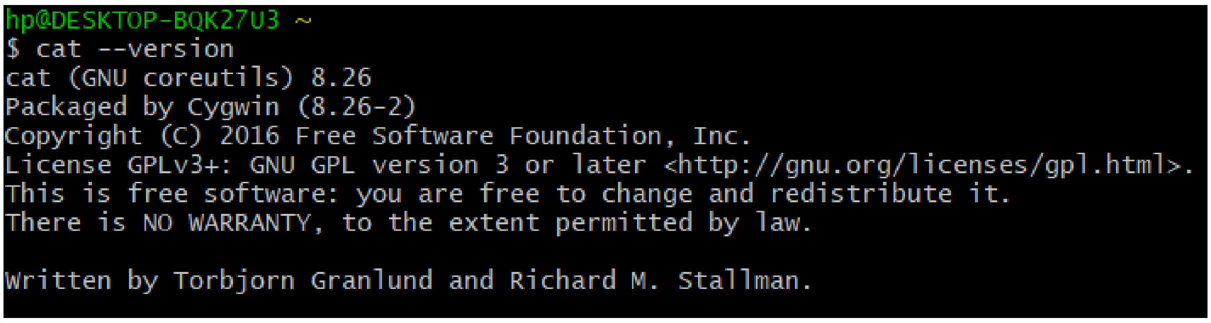
```
hp@DESKTOP-BQK27U3 ~
$ cat file1; cat file2
I am shaina mehta.
I live in delhi.
I am fine.
I am shaina mehta.
I live in delhi.
I am fine.
```



```
hp@DESKTOP-BQK27U3 ~
$ cat file2 -n
     1  I am shaina mehta.
     2  I live in delhi.
     3  I am fine.
```



```
hp@DESKTOP-BQK27U3 ~
$ cat -n file1
     1  I am shaina mehta.
     2  I live in delhi.
     3  I am fine.
```



```
hp@DESKTOP-BQK27U3 ~
$ cat --version
cat (GNU coreutils) 8.26
Packaged by Cygwin (8.26-2)
Copyright (C) 2016 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

Written by Torbjorn Granlund and Richard M. Stallman.
```

Result: Basic Linux commands has been executed successfully.

Experiment 2

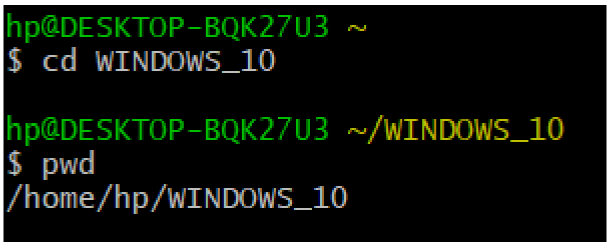
Date: 24-12-2020

Aim: To explore file and directory related commands.

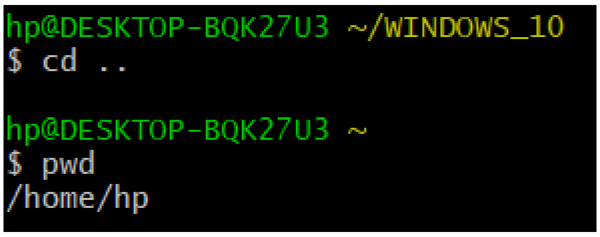
Software Used: Cgywin64 Terminal.

Theory:

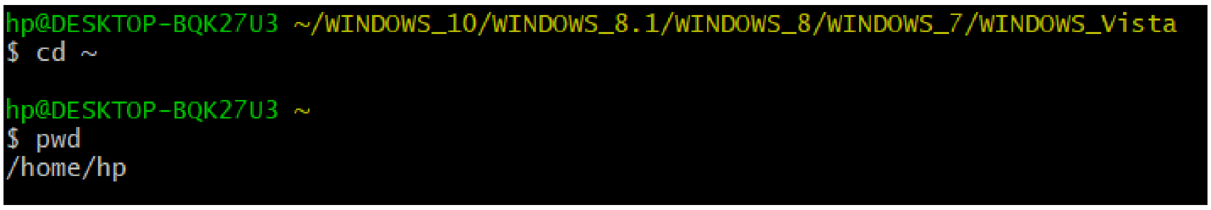
1. **cd:** cd command is known as change directory command. It is used to change the current working directory.
   Syntax: cd <name of the file/ commands/options>
   Note: Some symbols used in the form of command/options are:
   - **~ :** Specifies the location of your home directory.
   - **.. :** Specifies the location of the parent directory.



```
hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ pwd
/home/hp/WINDOWS_10
```



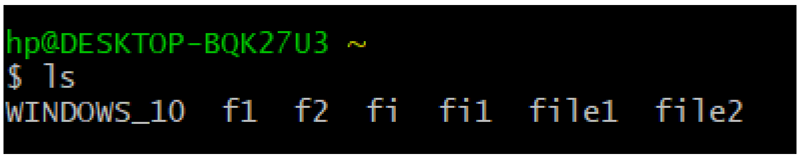
```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd ..

hp@DESKTOP-BQK27U3 ~
$ pwd
/home/hp
```



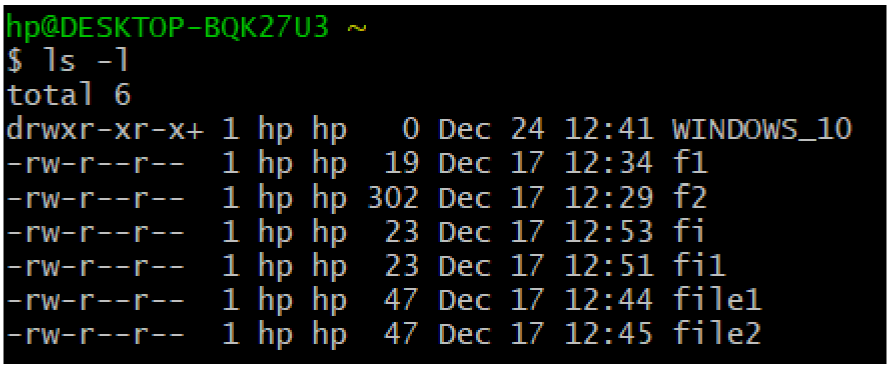
```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista
$ cd ~

hp@DESKTOP-BQK27U3 ~
$ pwd
/home/hp
```

2. **ls:** ls command is used to list the files in the current directory use.
   Syntax: ls <options> or ls
   Note:
   - Options used in ls command are:
     - **-l:** uses a long list format.
     - **-t:** sort by modification time, newest first.
     - **-r, --reverse:** reverse the order while sorting.
     - **-R, --recursive:** list subdirectories recursively.
     - **-i, --inode:** print the index number of each file.
     - *** :** can be used as a wildcard in UNIX/LINUX.
   - Options can be combined: ls -ltr.
   - **For Example:**

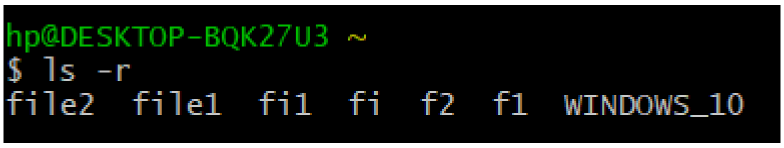
- **ls- lt:** list the files in time in reverse order with long.



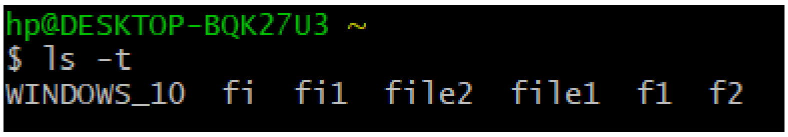
```
hp@DESKTOP-BQK27U3 ~
$ ls
WINDOWS_10  f1  f2  fi  fi1  file1  file2
```



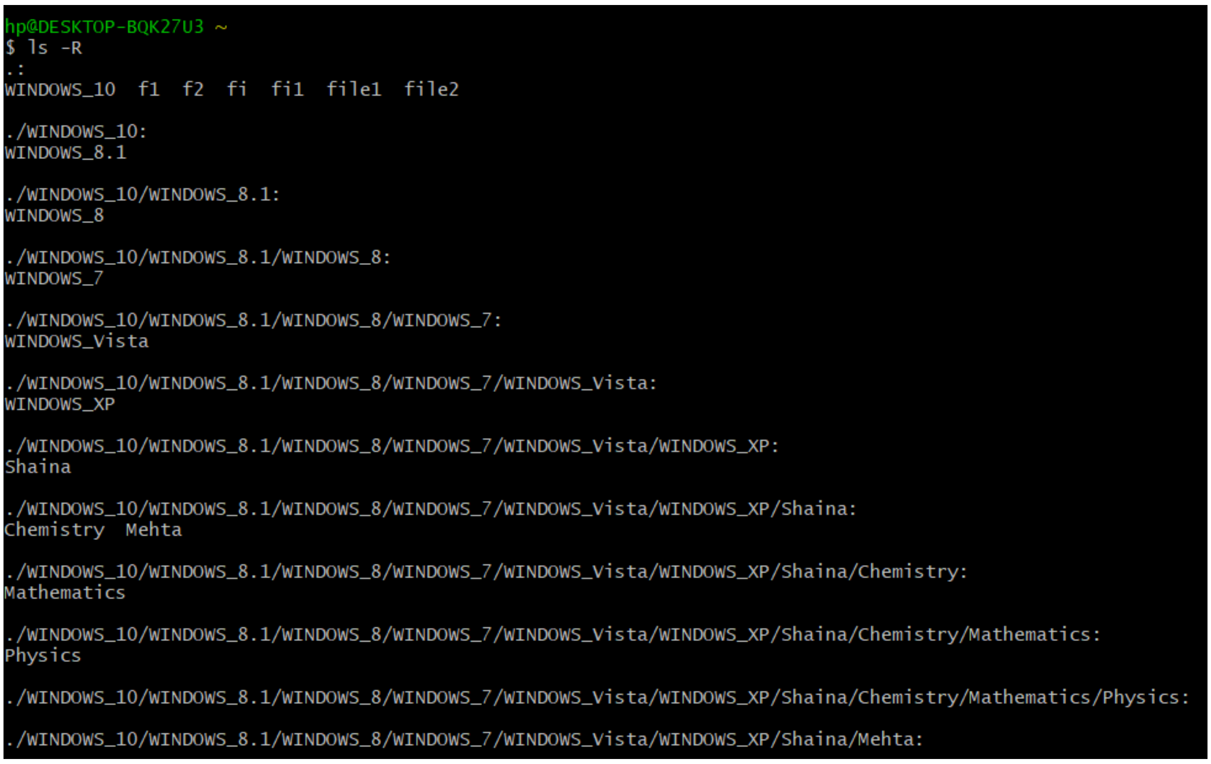
```
hp@DESKTOP-BQK27U3 ~
$ ls -l
total 6
drwxr-xr-x+ 1 hp hp   0 Dec 24 12:41 WINDOWS_10
-rw-r--r--  1 hp hp  19 Dec 17 12:34 f1
-rw-r--r--  1 hp hp 302 Dec 17 12:29 f2
-rw-r--r--  1 hp hp  23 Dec 17 12:53 fi
-rw-r--r--  1 hp hp  23 Dec 17 12:51 fi1
-rw-r--r--  1 hp hp  47 Dec 17 12:44 file1
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
```



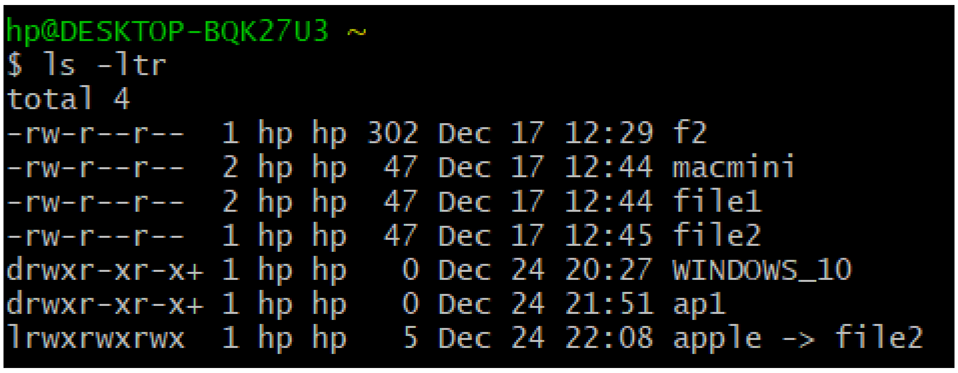
```
hp@DESKTOP-BQK27U3 ~
$ ls -r
file2  file1  fi1  fi  f2  f1  WINDOWS_10
```



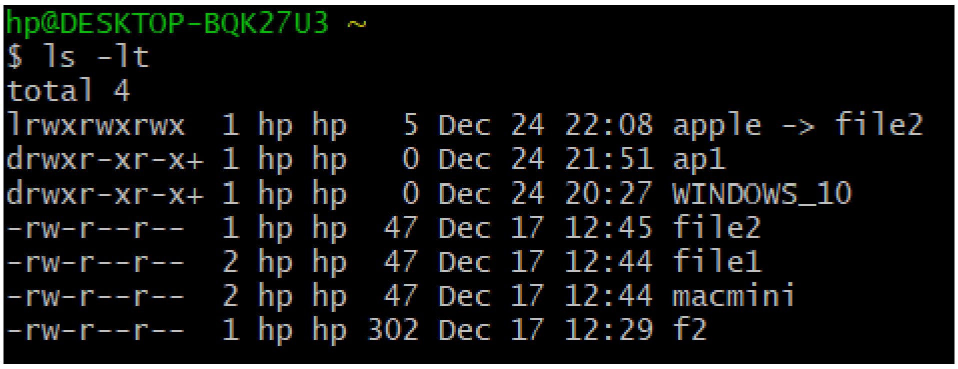
```
hp@DESKTOP-BQK27U3 ~
$ ls -t
WINDOWS_10  fi  fi1  file2  file1  f1  f2
```



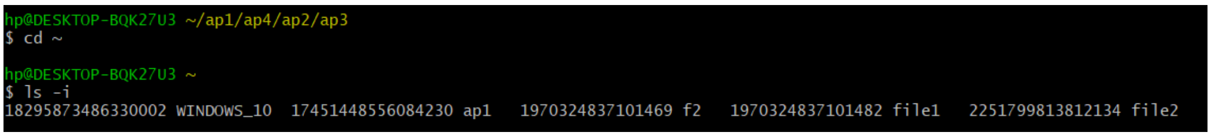
```
hp@DESKTOP-BQK27U3 ~
$ ls -R
.:
WINDOWS_10  f1  f2  fi  fi1  file1  file2

./WINDOWS_10:
WINDOWS_8.1

./WINDOWS_10/WINDOWS_8.1:
WINDOWS_8

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8:
WINDOWS_7

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7:
WINDOWS_Vista

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista:
WINDOWS_XP

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP:
Shaina

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina:
Chemistry  Mehta

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Chemistry:
Mathematics

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Chemistry/Mathematics:
Physics

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Chemistry/Mathematics/Physics:

./WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta:
```



```
hp@DESKTOP-BQK27U3 ~
$ ls -ltr
total 4
-rw-r--r--  1 hp hp 302 Dec 17 12:29 f2
-rw-r--r--  2 hp hp  47 Dec 17 12:44 macmini
-rw-r--r--  2 hp hp  47 Dec 17 12:44 file1
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
drwxr-xr-x+ 1 hp hp   0 Dec 24 20:27 WINDOWS_10
drwxr-xr-x+ 1 hp hp   0 Dec 24 21:51 ap1
lrwxrwxrwx  1 hp hp   5 Dec 24 22:08 apple -> file2
```

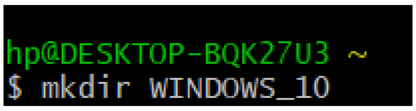


```
hp@DESKTOP-BQK27U3 ~
$ ls -lt
total 4
lrwxrwxrwx  1 hp hp   5 Dec 24 22:08 apple -> file2
drwxr-xr-x+ 1 hp hp   0 Dec 24 21:51 ap1
drwxr-xr-x+ 1 hp hp   0 Dec 24 20:27 WINDOWS_10
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
-rw-r--r--  2 hp hp  47 Dec 17 12:44 file1
-rw-r--r--  2 hp hp  47 Dec 17 12:44 macmini
-rw-r--r--  1 hp hp 302 Dec 17 12:29 f2
```

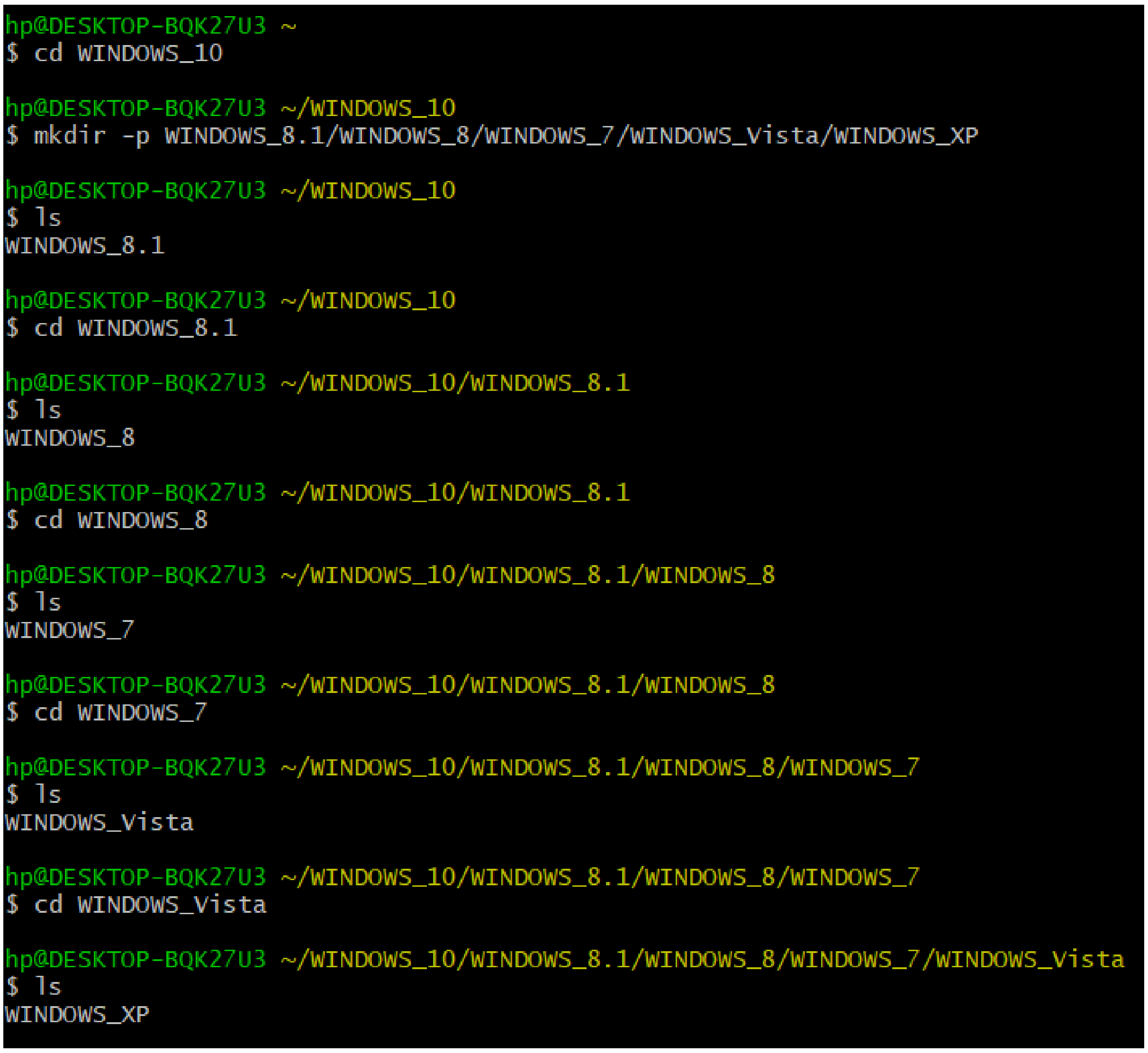


```
hp@DESKTOP-BQK27U3 ~/ap1/ap4/ap2/ap3
$ cd ~

hp@DESKTOP-BQK27U3 ~
$ ls -i
18295873486330002 WINDOWS_10  17451448556084230 ap1   1970324837101469 f2   1970324837101482 file1   2251799813812134 file2
```

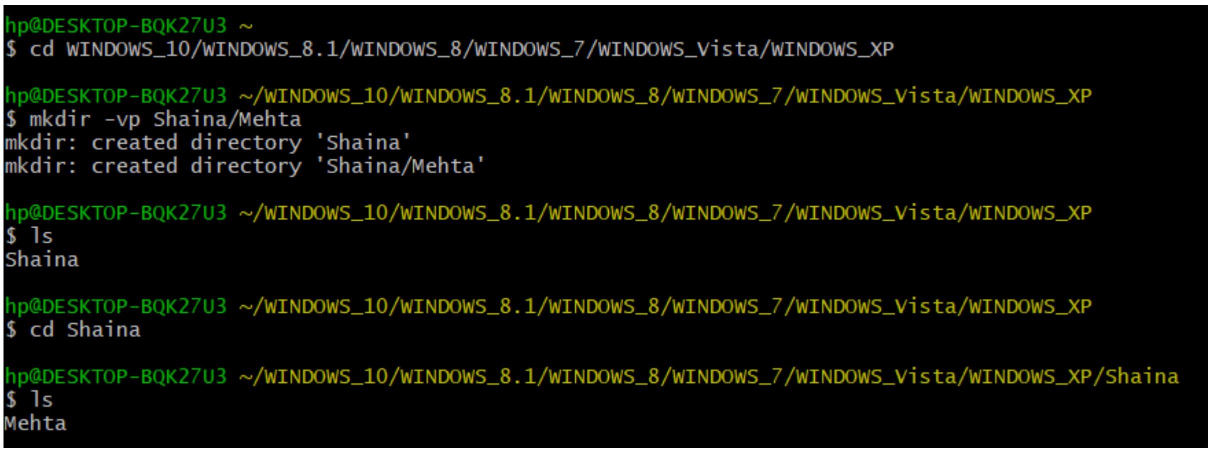
3. **mkdir:** mkdir command is used to create a new directory.
   Syntax: mkdir <option> <directory> or mkdir <directory>
   Note:
   - The command takes more than one directory name as its arguments.
   - Options used in mkdir command are:
     - ♦ **-m, --mode:** to set a file mode.
     - ♦ **-p, -parents:** no error if existing, otherwise make parent directory as needed.
     - ♦ **-v, --verbose:** print the message for each created directory.
     - ♦ **-z:** set SELinux security context for each created directory to the default type.
     - ♦ **context [=CTX]:** like -z, or if CTX is specified then set the SELinux or SMACK security to CTX.
     - ♦ **--help:** display the help and exit.
     - ♦ **--version:** output version information and exit.



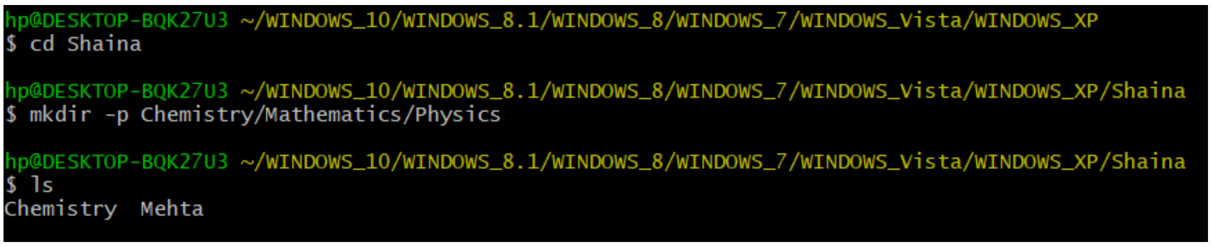
```
hp@DESKTOP-BQK27U3 ~
$ mkdir WINDOWS_10
```



```
hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ mkdir -p WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
WINDOWS_8.1

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd WINDOWS_8.1

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1
$ ls
WINDOWS_8

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1
$ cd WINDOWS_8

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8
$ ls
WINDOWS_7

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8
$ cd WINDOWS_7

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7
$ ls
WINDOWS_Vista

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7
$ cd WINDOWS_Vista

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista
$ ls
WINDOWS_XP
```

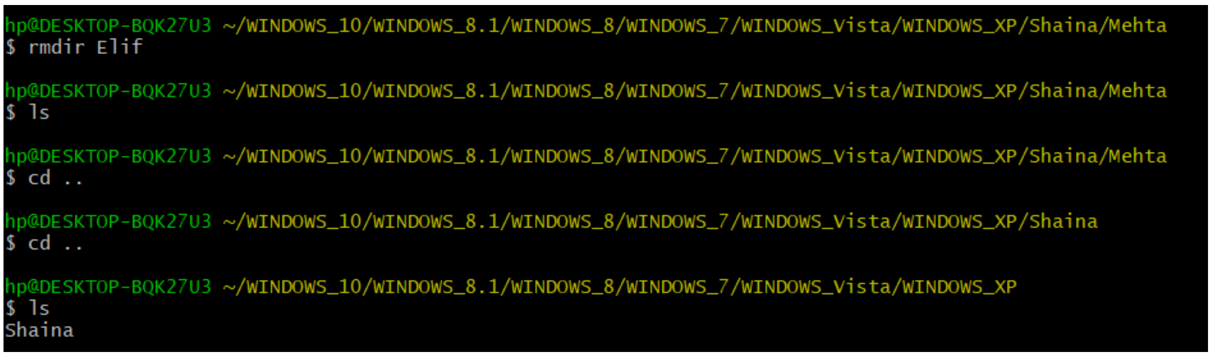


```
hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ mkdir -vp Shaina/Mehta
mkdir: created directory 'Shaina'
mkdir: created directory 'Shaina/Mehta'

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ ls
Shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cd Shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina
$ ls
Mehta
```



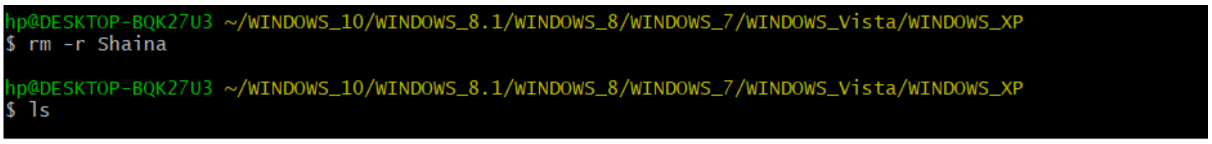
```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cd Shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina
$ mkdir -p Chemistry/Mathematics/Physics

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina
$ ls
Chemistry  Mehta
```

4. **rmdir:** rmdir command is used to remove empty directories.
   Syntax: rmdir <option> <directory> or rmdir <option>
   Note: Options used in rmdir are:
   - **--ignore-fail-on-non-empty:** ignore each failure that is solely because a directory is non - empty.
   - **-p, --parents:** remove directory and its ancestors. For example: 'rmdir -p a/b/c' is similar to 'a/b/c a/b a'.
   - **-v, -verbose:** outputs a diagnostic for every directory processed
   - **--help:** displays the help and exit.
   - **--version:** outputs the version information and exit.

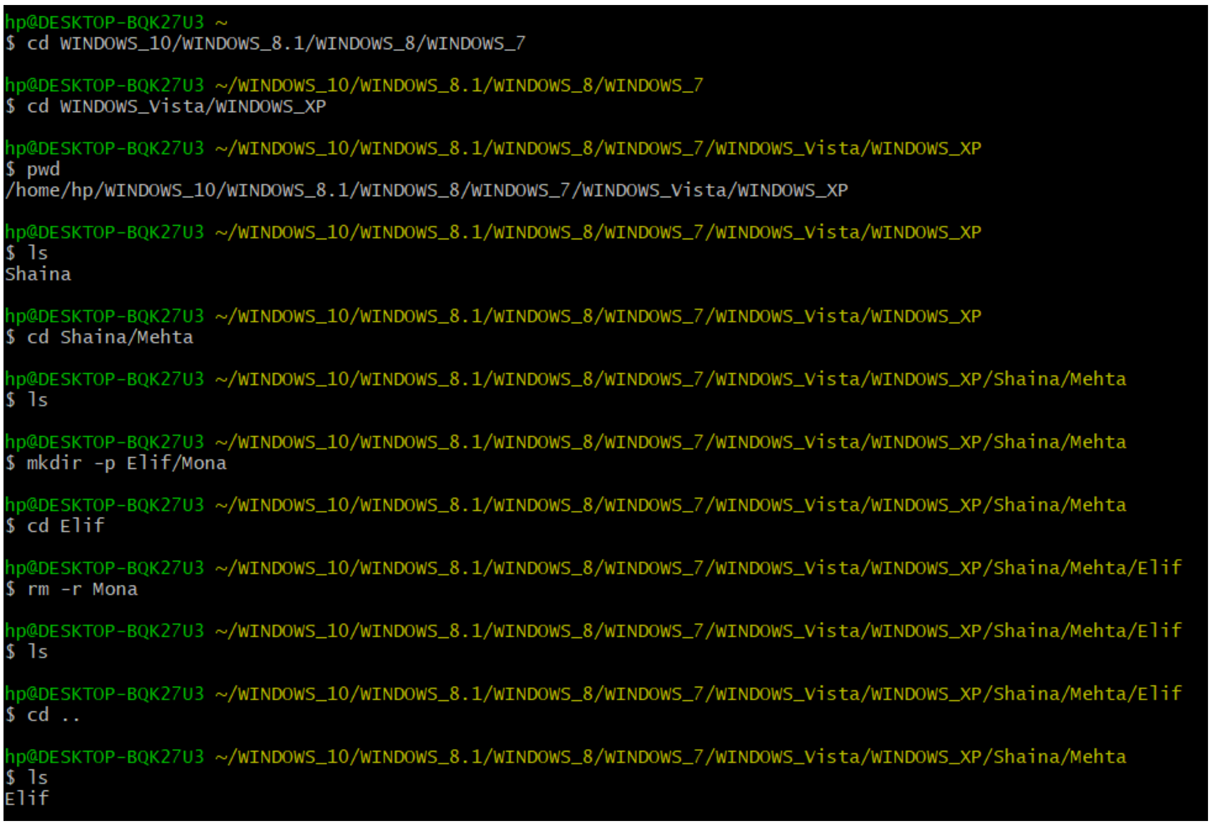


```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ rmdir Elif

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ ls

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ cd ..

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina
$ cd ..

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ ls
Shaina
```

5. **rm:** rm command is used to remove a file.
   Syntax: rm <directory> or rm <options> <directory>
   Note: Options used in rm command are:
   - **-f, --force:** ignore non existing files and arguments and never prompt.
   - **-i:** prompt before every removal.
   - **-I:** prompt once before removing more than one files, or when removing recursively; less intuitive than -i, while still giving protection against more mistakes.
   - **--interactive [=WHEN]:** prompt according to WHEN: never, once (-I), or always (-i); without WHEN, prompt always.
   - **--one – file - system:** when removing the hierarchy recursively, skip any directory that is on the file system different from that corresponding command line argument.
   - **--no - preserve – root:** do not treat '/' specially.
   - **--preserve – root:** do not remove '/' (default).
   - **-r, -R, --recursive:** remove directories and their contents recursively.
   - **-d, --dir:** remove empty directories.
   - **-v, -verbose:** explain what is being done.
   - **--help:** display this help and exit.
   - **--version:** output version information and exit.



```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ rm -r Shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ ls
```

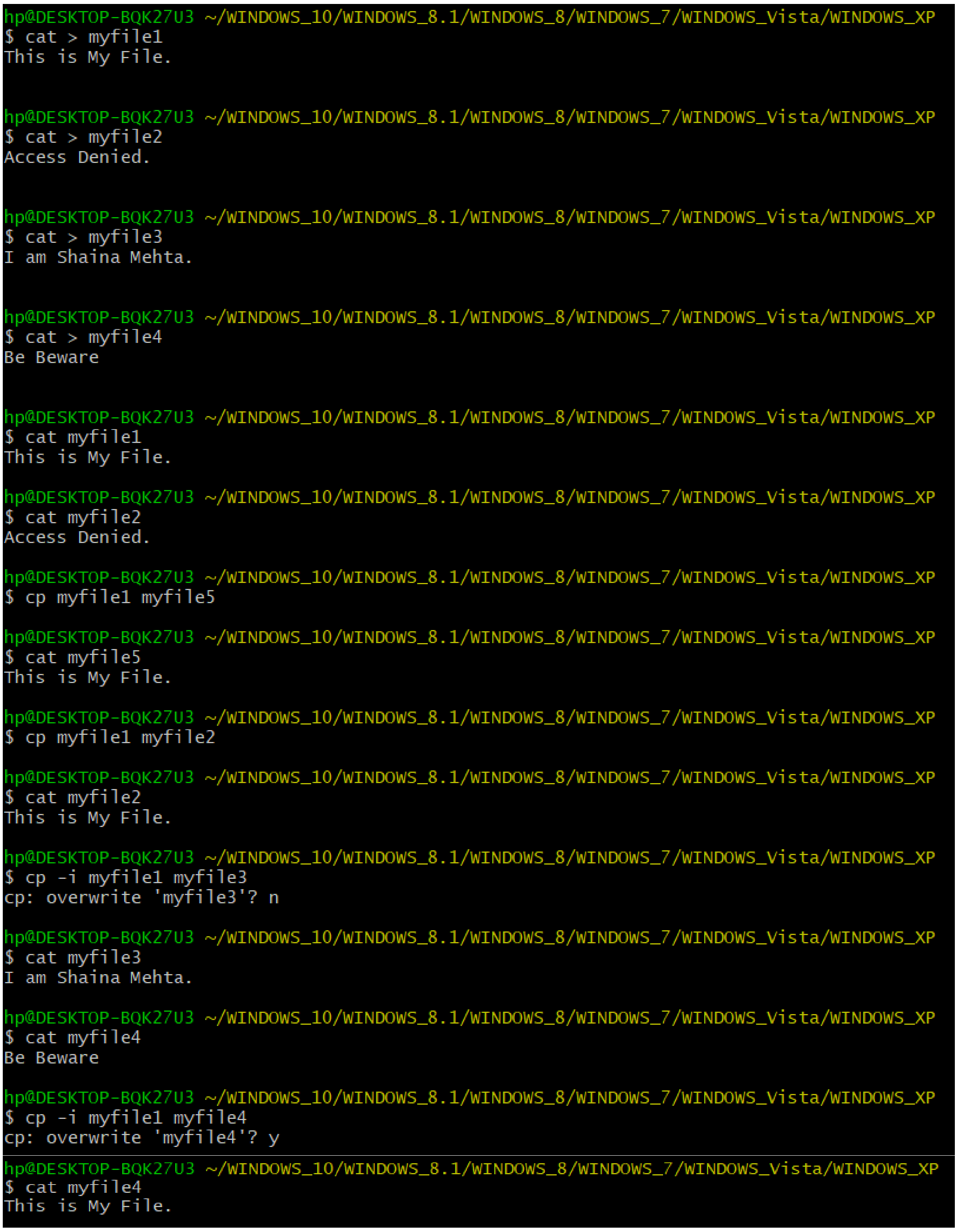


```
hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7
$ cd WINDOWS_Vista/WINDOWS_XP

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ pwd
/home/hp/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ ls
Shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cd Shaina/Mehta

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ ls

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ mkdir -p Elif/Mona

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ cd Elif

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta/Elif
$ rm -r Mona

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta/Elif
$ ls

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta/Elif
$ cd ..

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP/Shaina/Mehta
$ ls
Elif
```

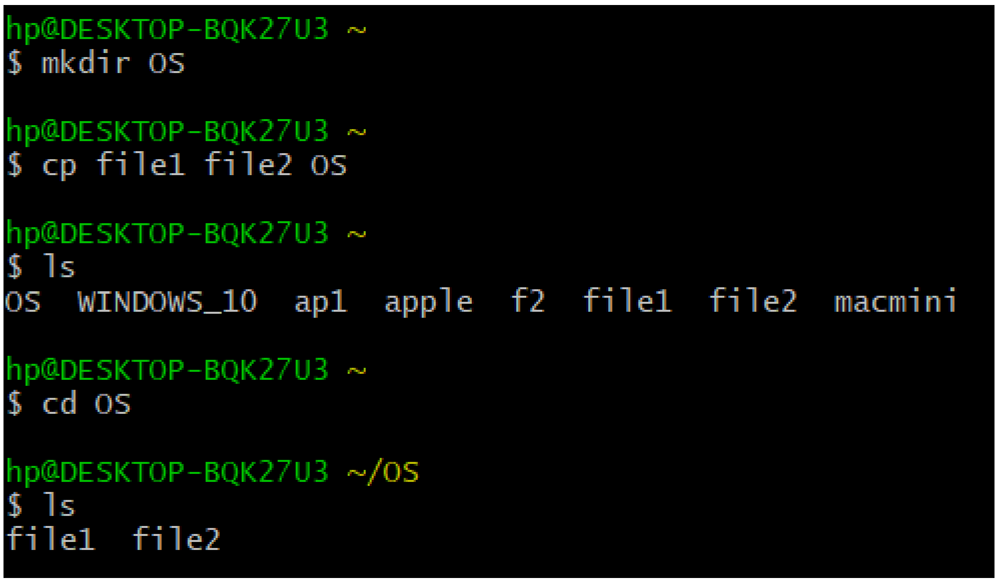
6. **cp:** cp command is used to copy the files or group of files or directories.

   Syntax:
   - cp <source file> <destination file>: copy the contents of one file to another.
   - cp <file1> <file2> <directory name>: copy multiple files in a directory.
   - cp -i <source file> <destination file>: asks the user whether to copy the source file to the destination file or not.

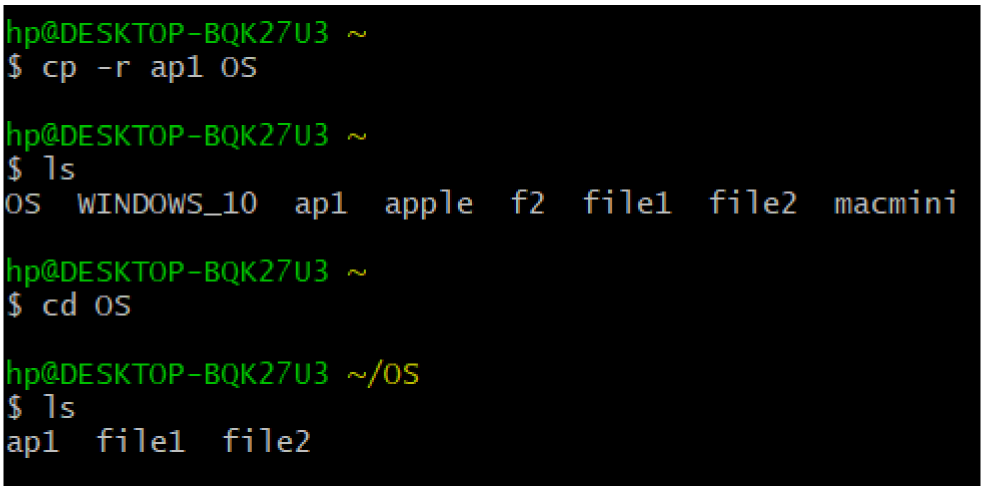
   Note:
   - By default, the cp command will not copy directories. Attempting to copy a directory results in an error.
   - To copy a directory, pass the -R or -r or –recursive flag. This will recursively copy a folder and create a copy.
   - **Syntax:** Cp –r <source directory> <destination directory>



```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat > myfile1
This is My File.


hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat > myfile2
Access Denied.


hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat > myfile3
I am Shaina Mehta.


hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat > myfile4
Be Beware


hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile1
This is My File.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile2
Access Denied.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cp myfile1 myfile5

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile5
This is My File.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cp myfile1 myfile2

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile2
This is My File.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cp -i myfile1 myfile3
cp: overwrite 'myfile3'? n

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile3
I am Shaina Mehta.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile4
Be Beware

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cp -i myfile1 myfile4
cp: overwrite 'myfile4'? y

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cat myfile4
This is My File.
```

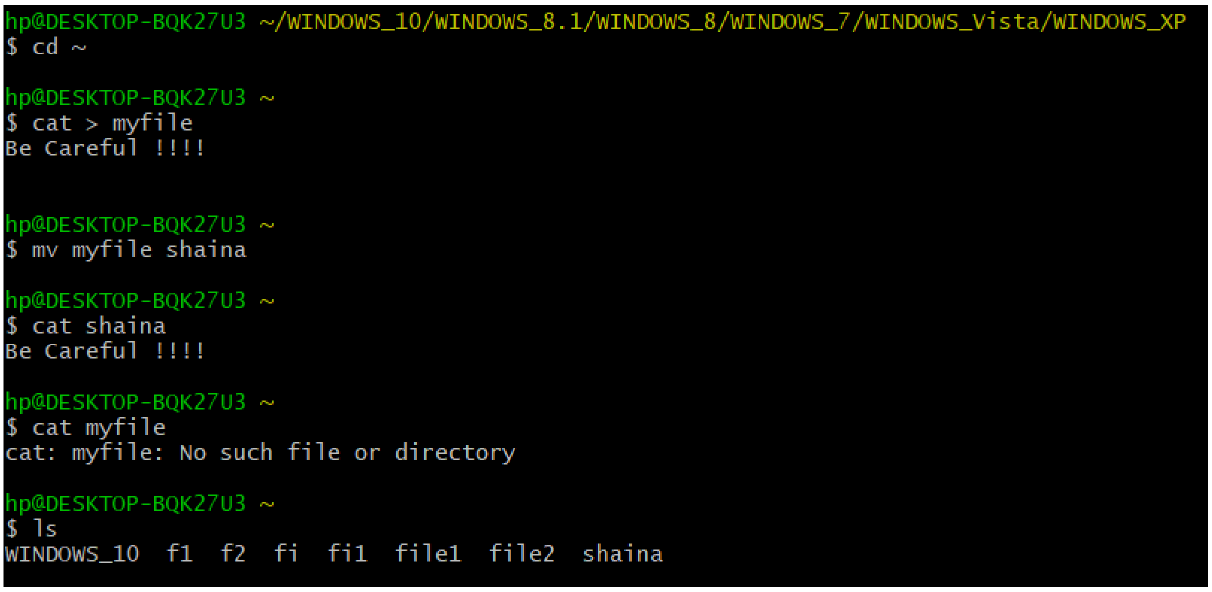


```
hp@DESKTOP-BQK27U3 ~
$ mkdir OS

hp@DESKTOP-BQK27U3 ~
$ cp file1 file2 OS

hp@DESKTOP-BQK27U3 ~
$ ls
OS  WINDOWS_10  ap1  apple  f2  file1  file2  macmini

hp@DESKTOP-BQK27U3 ~
$ cd OS

hp@DESKTOP-BQK27U3 ~/OS
$ ls
file1  file2
```

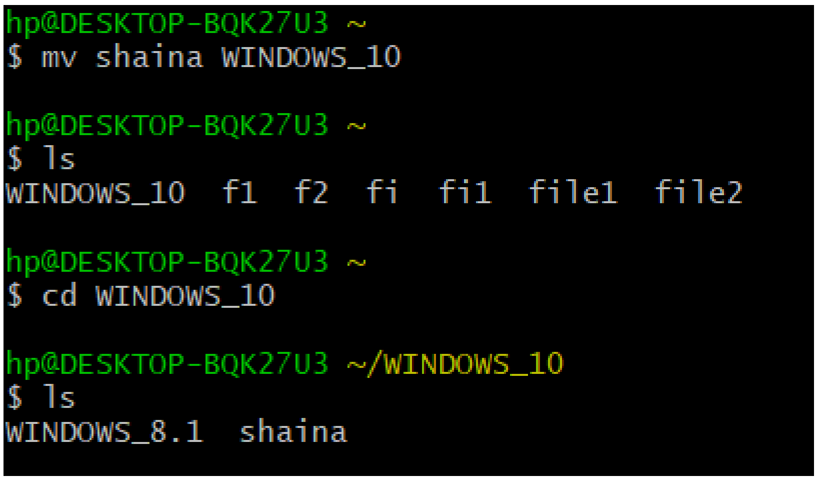


```
hp@DESKTOP-BQK27U3 ~
$ cp -r ap1 OS

hp@DESKTOP-BQK27U3 ~
$ ls
OS  WINDOWS_10  ap1  apple  f2  file1  file2  macmini

hp@DESKTOP-BQK27U3 ~
$ cd OS

hp@DESKTOP-BQK27U3 ~/OS
$ ls
ap1  file1  file2
```

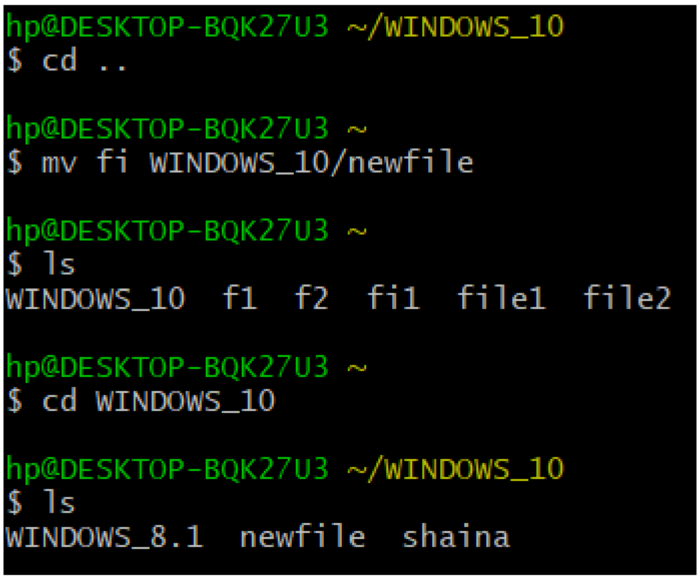
7. **mv:** mv command is used to move files or directories from one place to another.
   Syntax:
   - mv <old file> <new file>:
     - ♦ To move a file using the mv command pass the name of the file and then the new name for the file
     - ♦ **For Example:** mv file1 file2
     - ♦ In above example file1 is renamed to file2.
   - mv <old directory> <new directory>: to move a directory.
   - mv <file name> <directory name> or mv <file name> <directory name/ filename>: to move a file in a directory.
   - mv <file1> <file2> <file3> <directory name>: to move multiple files in a given directory.
   - mv -I file1 file2: prompt before overwriting a file.



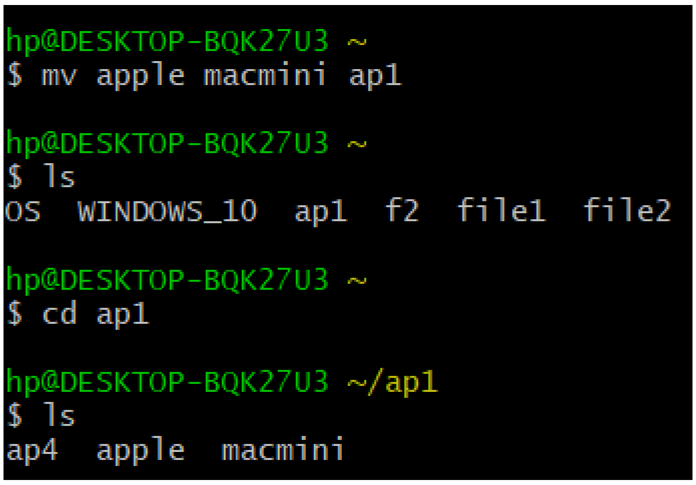
```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista/WINDOWS_XP
$ cd ~

hp@DESKTOP-BQK27U3 ~
$ cat > myfile
Be Careful !!!!


hp@DESKTOP-BQK27U3 ~
$ mv myfile shaina

hp@DESKTOP-BQK27U3 ~
$ cat shaina
Be Careful !!!!

hp@DESKTOP-BQK27U3 ~
$ cat myfile
cat: myfile: No such file or directory

hp@DESKTOP-BQK27U3 ~
$ ls
WINDOWS_10  f1  f2  fi  fi1  file1  file2  shaina
```



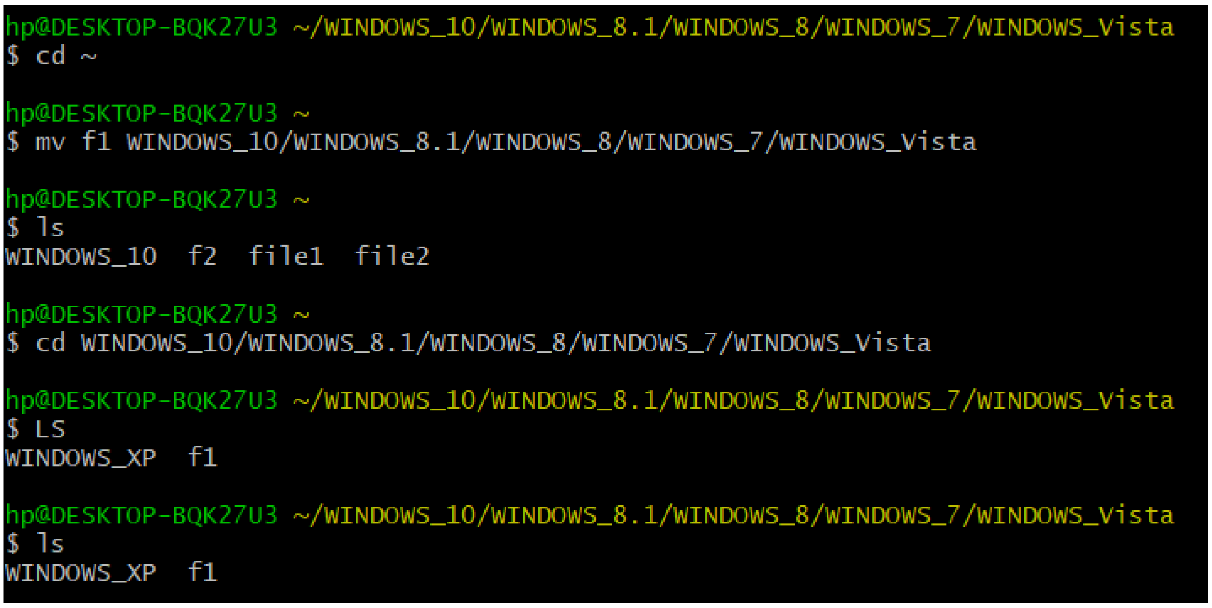
```
hp@DESKTOP-BQK27U3 ~
$ mv shaina WINDOWS_10

hp@DESKTOP-BQK27U3 ~
$ ls
WINDOWS_10  f1  f2  fi  fi1  file1  file2

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
WINDOWS_8.1  shaina
```



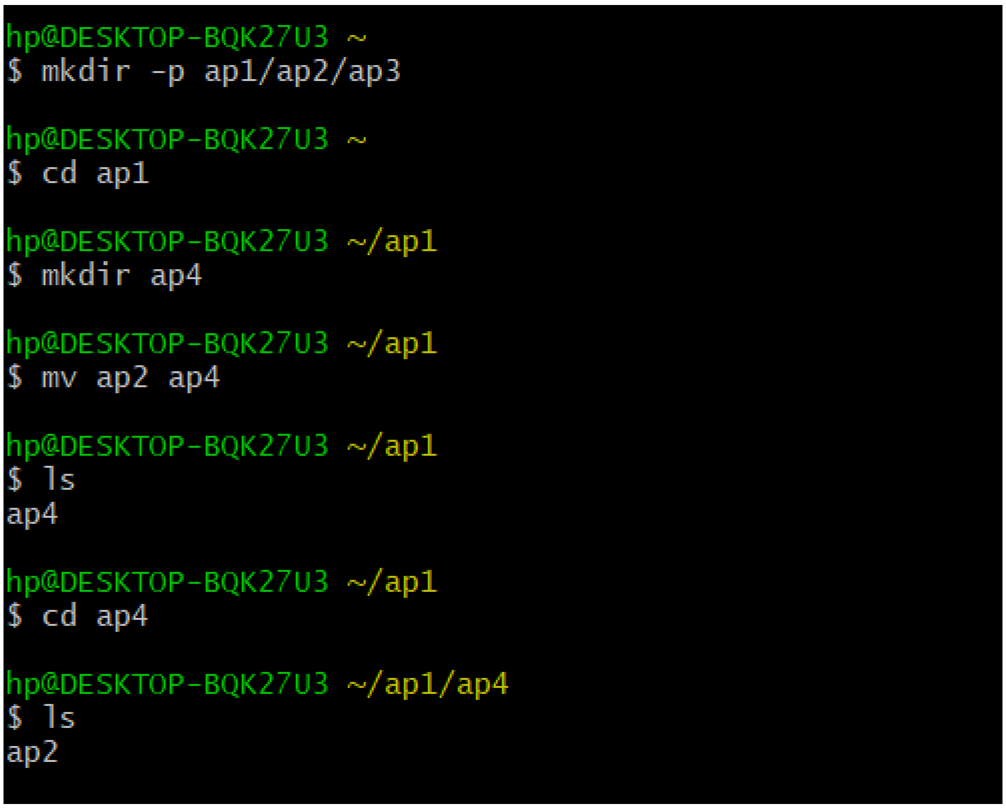
```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd ..

hp@DESKTOP-BQK27U3 ~
$ mv fi WINDOWS_10/newfile

hp@DESKTOP-BQK27U3 ~
$ ls
WINDOWS_10  f1  f2  fi1  file1  file2

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
WINDOWS_8.1  newfile  shaina
```



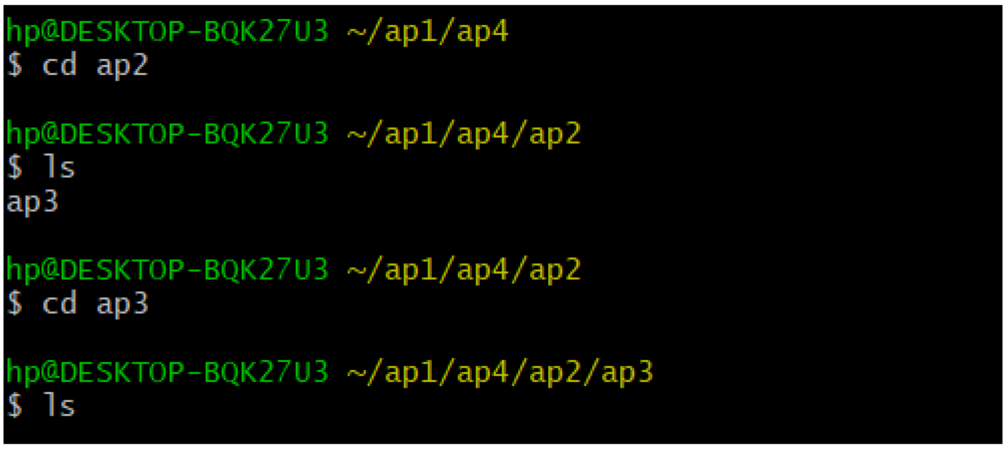
```
hp@DESKTOP-BQK27U3 ~
$ mv apple macmini ap1

hp@DESKTOP-BQK27U3 ~
$ ls
OS  WINDOWS_10  ap1  f2  file1  file2

hp@DESKTOP-BQK27U3 ~
$ cd ap1

hp@DESKTOP-BQK27U3 ~/ap1
$ ls
ap4  apple  macmini
```



```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista
$ cd ~

hp@DESKTOP-BQK27U3 ~
$ mv f1 WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista

hp@DESKTOP-BQK27U3 ~
$ ls
WINDOWS_10  f2  file1  file2

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista
$ LS
WINDOWS_XP  f1

hp@DESKTOP-BQK27U3 ~/WINDOWS_10/WINDOWS_8.1/WINDOWS_8/WINDOWS_7/WINDOWS_Vista
$ ls
WINDOWS_XP  f1
```

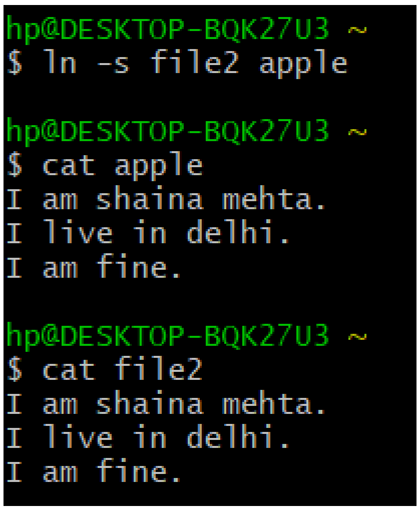
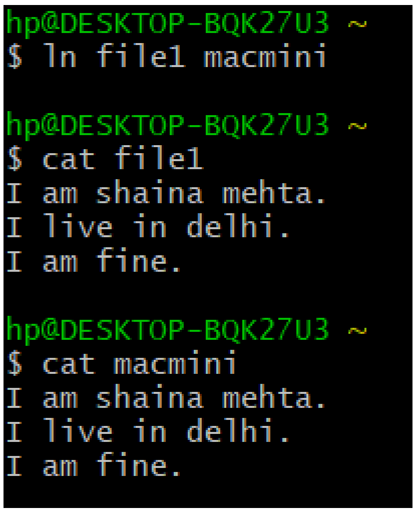


```
hp@DESKTOP-BQK27U3 ~
$ mkdir -p ap1/ap2/ap3

hp@DESKTOP-BQK27U3 ~
$ cd ap1

hp@DESKTOP-BQK27U3 ~/ap1
$ mkdir ap4

hp@DESKTOP-BQK27U3 ~/ap1
$ mv ap2 ap4

hp@DESKTOP-BQK27U3 ~/ap1
$ ls
ap4

hp@DESKTOP-BQK27U3 ~/ap1
$ cd ap4

hp@DESKTOP-BQK27U3 ~/ap1/ap4
$ ls
ap2
```



```
hp@DESKTOP-BQK27U3 ~/ap1/ap4
$ cd ap2

hp@DESKTOP-BQK27U3 ~/ap1/ap4/ap2
$ ls
ap3

hp@DESKTOP-BQK27U3 ~/ap1/ap4/ap2
$ cd ap3

hp@DESKTOP-BQK27U3 ~/ap1/ap4/ap2/ap3
$ ls
```

8. **ln:** ln command is used to create links between files.
   - **Soft Link:** ln –s filename soft_link name. -s makes symbolic links instead of hard link.
     Syntax: ln -s <file1> <file2>
   - **Hard link:** ln filename hard_link name.
     Syntax: ln <file1> <file2>


```
hp@DESKTOP-BQK27U3 ~
$ ln -s file2 apple

hp@DESKTOP-BQK27U3 ~
$ cat apple
I am shaina mehta.
I live in delhi.
I am fine.

hp@DESKTOP-BQK27U3 ~
$ cat file2
I am shaina mehta.
I live in delhi.
I am fine.
```

```
hp@DESKTOP-BQK27U3 ~
$ ln file1 macmini

hp@DESKTOP-BQK27U3 ~
$ cat file1
I am shaina mehta.
I live in delhi.
I am fine.

hp@DESKTOP-BQK27U3 ~
$ cat macmini
I am shaina mehta.
I live in delhi.
I am fine.
```

Result: Various file and directory related commands has been explored and executed successfully.

<u>Experiment 3</u>

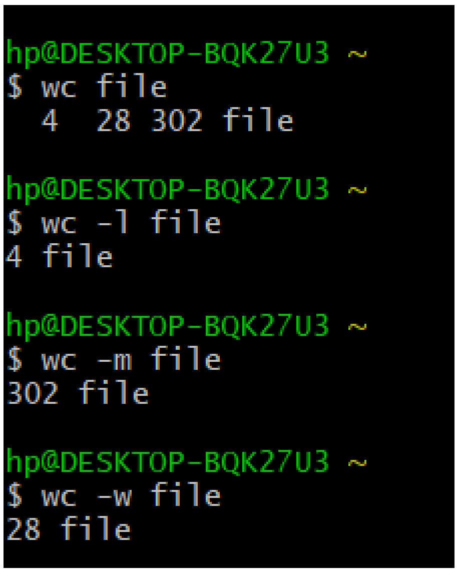
<u>Date:</u> 7 - 01 - 2021

<u>Aim:</u> To explore advance Linux commands.

<u>Software Used:</u> Cgywin64 Terminal.

<u>Theory:</u>

1. **wc:** wc command is used for printing newline, word and byte counts for files. It can return the number of lines in a file, the number of characters in a file and the number of words in a file. The output is number of lines, number of words, number of bytes, filename.
   Syntax: wc <filename> or wc <options> <filename>.
   Note: Options for wc command are:
   - -l - To print the number of lines in a file.
   - -m - To print the number of characters in a file.
   - -w - To print the number of words in a file.

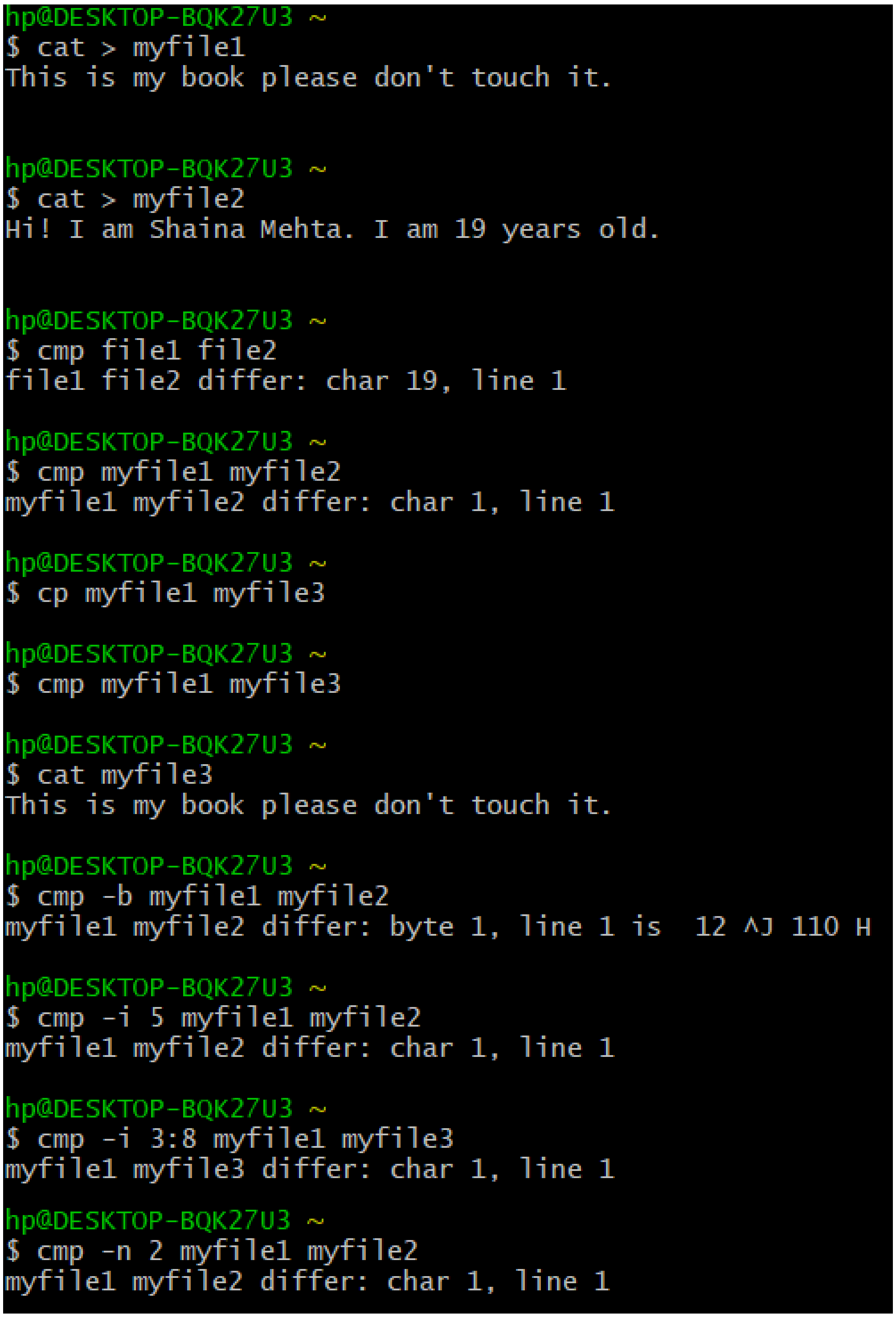


```
hp@DESKTOP-BQK27U3 ~
$ wc file
  4  28 302 file

hp@DESKTOP-BQK27U3 ~
$ wc -l file
4 file

hp@DESKTOP-BQK27U3 ~
$ wc -m file
302 file

hp@DESKTOP-BQK27U3 ~
$ wc -w file
28 file
```

2. **cmp:** cmp command is used to compare the two files byte by byte. It helps you to find out whether the two files are identical or not. It reports the location of the first mismatch to the screen if difference is found and if no difference is found i.e the files compared are identical. It displays no message and simply returns the prompt if the files compared are identical.
   Syntax: cmp <option> <filename> or cmp <filename>.
   Note:
   - -b - display the differing bytes in its output.
     cmp -b file1 file2
   - -i [bytes-to-be-skipped] - Now, this option when used with cmp command helps to skip a particular number of initial bytes from both the files and then after skipping it compares the files.
     cmp –i 5 file1 file2

- -i [bytes to be skipped from first file] : [bytes to be skipped from second file] - This option is very much similar to the above -i [bytes to be skipped] option but with the difference that now it allows us to input the number of bytes we want to skip from both the files separately.

  ```
  cmp –i 4:4 file1 file2
  ```
- -n [number of bytes to be compared] option - This option allows you to limit the number of bytes you want to compare, like if there is only need to compare at most 25 or 50 bytes.



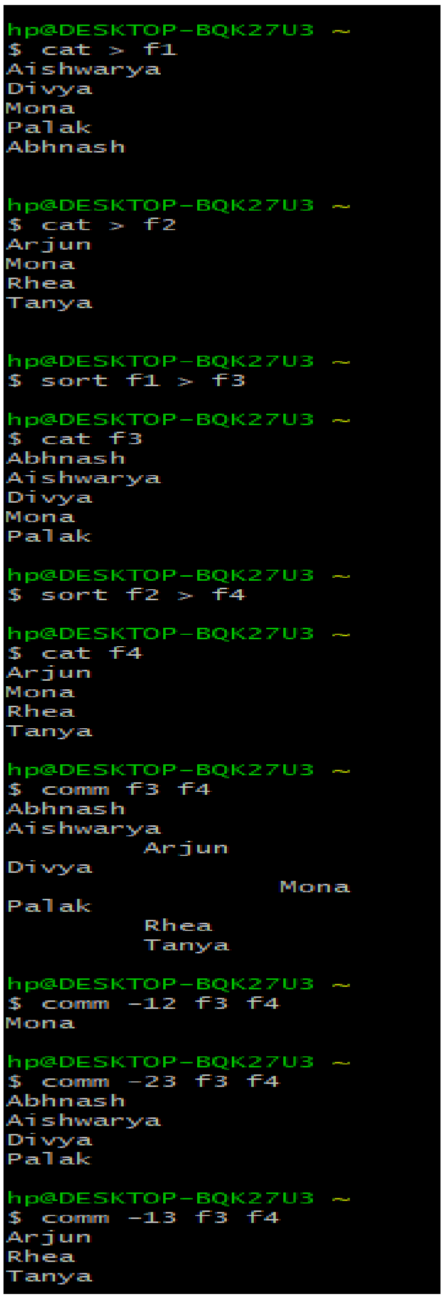
```
hp@DESKTOP-BQK27U3 ~
$ cat > myfile1
This is my book please don't touch it.


hp@DESKTOP-BQK27U3 ~
$ cat > myfile2
Hi! I am Shaina Mehta. I am 19 years old.


hp@DESKTOP-BQK27U3 ~
$ cmp file1 file2
file1 file2 differ: char 19, line 1

hp@DESKTOP-BQK27U3 ~
$ cmp myfile1 myfile2
myfile1 myfile2 differ: char 1, line 1

hp@DESKTOP-BQK27U3 ~
$ cp myfile1 myfile3

hp@DESKTOP-BQK27U3 ~
$ cmp myfile1 myfile3

hp@DESKTOP-BQK27U3 ~
$ cat myfile3
This is my book please don't touch it.

hp@DESKTOP-BQK27U3 ~
$ cmp -b myfile1 myfile2
myfile1 myfile2 differ: byte 1, line 1 is  12 ^J 110 H

hp@DESKTOP-BQK27U3 ~
$ cmp -i 5 myfile1 myfile2
myfile1 myfile2 differ: char 1, line 1

hp@DESKTOP-BQK27U3 ~
$ cmp -i 3:8 myfile1 myfile3
myfile1 myfile3 differ: char 1, line 1

hp@DESKTOP-BQK27U3 ~
$ cmp -n 2 myfile1 myfile2
myfile1 myfile2 differ: char 1, line 1
```

3. **comm:**
   - It requires two sorted files and lists the differing entries in different columns.
   - When you run comm, it displays a three – columnar output.
   - The first column contains the lines unique to the first file, and the second column shows the lines unique to the second file. The third column displays lines to both files.

Syntax: comm <options> <sorted_file_1> <sorted_file_2> or comm <sorted_file_1> <sorted_file_2>.

Note:

- These commands require single column output from comm, and comm can produce using the options -1, -2 or -3.
- To drop a particular column simply use its column number as an option prefix.



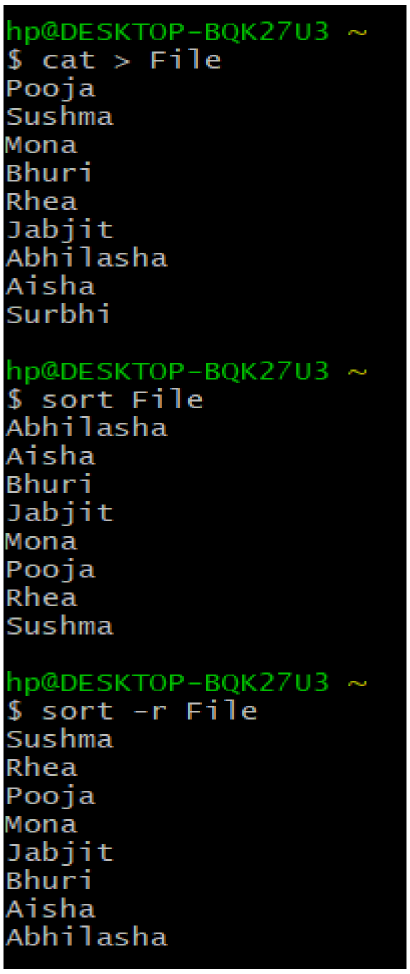
```
hp@DESKTOP-BQK27U3 ~
$ cat > f1
Aishwarya
Divya
Mona
Palak
Abhnash

hp@DESKTOP-BQK27U3 ~
$ cat > f2
Arjun
Mona
Rhea
Tanya

hp@DESKTOP-BQK27U3 ~
$ sort f1 > f3

hp@DESKTOP-BQK27U3 ~
$ cat f3
Abhnash
Aishwarya
Divya
Mona
Palak

hp@DESKTOP-BQK27U3 ~
$ sort f2 > f4

hp@DESKTOP-BQK27U3 ~
$ cat f4
Arjun
Mona
Rhea
Tanya

hp@DESKTOP-BQK27U3 ~
$ comm f3 f4
Abhnash
Aishwarya
        Arjun
Divya
                Mona
Palak
        Rhea
        Tanya

hp@DESKTOP-BQK27U3 ~
$ comm -12 f3 f4
Mona

hp@DESKTOP-BQK27U3 ~
$ comm -23 f3 f4
Abhnash
Aishwarya
Divya
Palak

hp@DESKTOP-BQK27U3 ~
$ comm -13 f3 f4
Arjun
Rhea
Tanya
```

4. **sort:**
   - sort lines alphabetically by default.
   - Running sort filename writes the contents of the filename in alphabetical order to standard output.

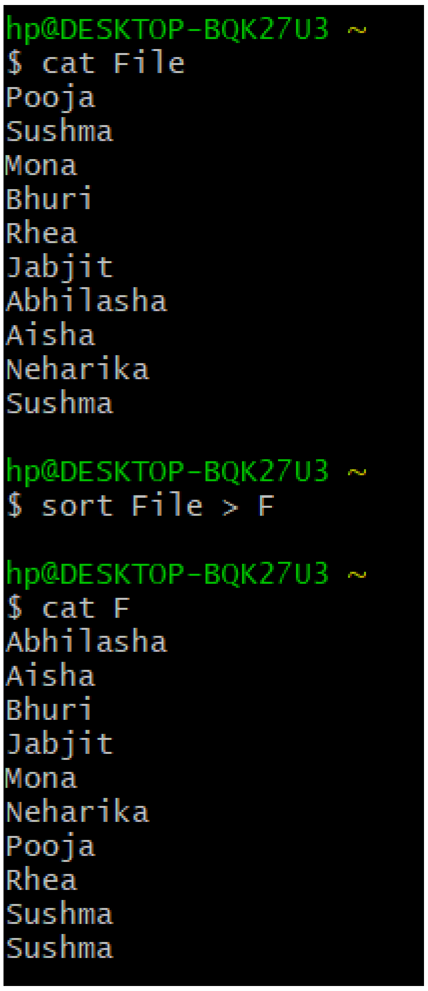
Syntax: sort <options> <filename> or sort <filename> or sort <file1> <file2>

Note:

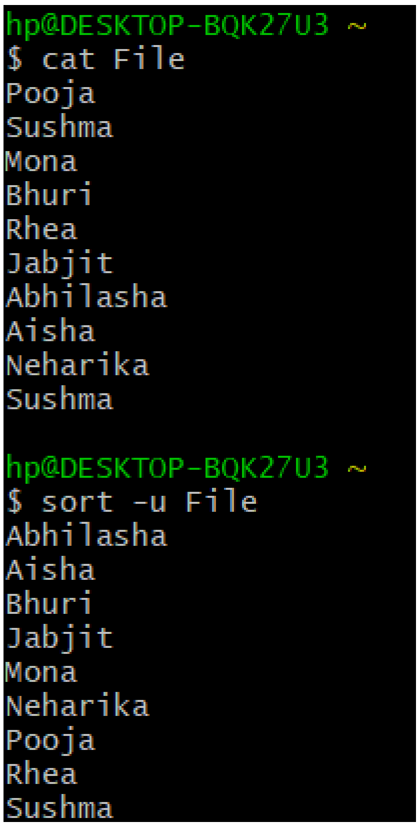
- -r - sort in reverse order and write the result to standard output.
- -n - This will sort from lowest number to highest number and write the result to standard output.
- To sort and remove duplicates pass the -u option to sort. This will write a sorted list to standard output and remove duplicates.
- To sort by month pass the -M option to sort.



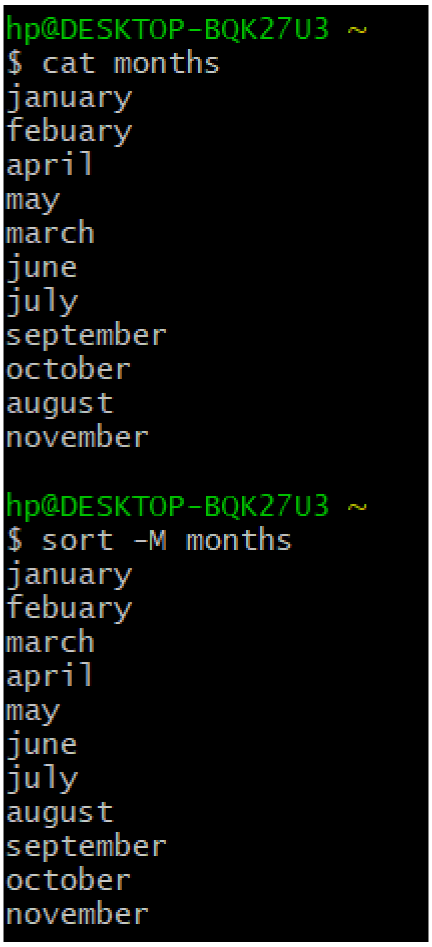
```
hp@DESKTOP-BQK27U3 ~
$ cat > File
Pooja
Sushma
Mona
Bhuri
Rhea
Jabjit
Abhilasha
Aisha
Surbhi

hp@DESKTOP-BQK27U3 ~
$ sort File
Abhilasha
Aisha
Bhuri
Jabjit
Mona
Pooja
Rhea
Sushma

hp@DESKTOP-BQK27U3 ~
$ sort -r File
Sushma
Rhea
Pooja
Mona
Jabjit
Bhuri
Aisha
Abhilasha
```



```
hp@DESKTOP-BQK27U3 ~
$ cat File
Pooja
Sushma
Mona
Bhuri
Rhea
Jabjit
Abhilasha
Aisha
Neharika
Sushma

hp@DESKTOP-BQK27U3 ~
$ sort File > F

hp@DESKTOP-BQK27U3 ~
$ cat F
Abhilasha
Aisha
Bhuri
Jabjit
Mona
Neharika
Pooja
Rhea
Sushma
Sushma
```



```
hp@DESKTOP-BQK27U3 ~
$ cat File
Pooja
Sushma
Mona
Bhuri
Rhea
Jabjit
Abhilasha
Aisha
Neharika
Sushma

hp@DESKTOP-BQK27U3 ~
$ sort -u File
Abhilasha
Aisha
Bhuri
Jabjit
Mona
Neharika
Pooja
Rhea
Sushma
```



```
hp@DESKTOP-BQK27U3 ~
$ cat months
january
febuary
april
may
march
june
july
september
october
august
november

hp@DESKTOP-BQK27U3 ~
$ sort -M months
january
febuary
march
april
may
june
july
august
september
october
november
```

5. **Creating Files in Linux:** It requires the use of an Editor.Various editors are used for this purpose which are:
    - nano / pico
    - vi
    - emacs

    Vi Editor:
    - The VI editor is the most popular and classic text editor in the Linux family.
    - Below, are some reasons which make it a widely used editor:
        - ♦ available in almost all Linux Distributions
        - ♦ works the same across different platforms and Distributions
        - ♦ user-friendly

    Modes of Vi Editor:
    - Command Mode
    - Insert Mode
    - Escape Mode

    Command Mode:
    - vi starts in Command Mode.
    - vi interprets any characters we type as commands and does not display them in the window.
    - This mode allows us to move through a file, and to delete, copy, or paste a piece of text.

- To enter into Command Mode from any other mode, it requires pressing the [Esc] key. If we press [Esc] when we are already in Command Mode, then vi will beep or flash the screen.

Insert Mode:

- Enables you to insert text into the file.
- Everything that’s typed in this mode is interpreted as input and finally, it is put in the file.
- The vi always starts in command mode. To enter text, you must be in insert mode. To come in insert mode, you simply type i. To get out of insert mode, press the Esc key, which will put you back into command mode.

Escape Mode:

- enables you to perform tasks such as saving files, executing commands.
- invoked by typing a colon [:], while vi is in Command Mode.
- The cursor will jump to the last line of the screen and vi will wait for a command.

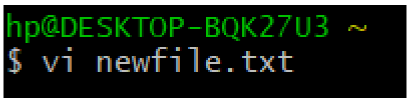
Vi Editor Commands:

- i - Insert at cursor (goes into insert mode)
- a - Write after cursor (goes into insert mode)
- A - Write at the end of line (goes into insert mode)
- ESC - Terminate insert mode
- U - Undo all changes to the entire line
- - Open a new line (goes into insert mode)
- dd - Delete line
- 3dd - Delete 3 lines.
- D - Delete contents of line after the cursor
- dw - Delete word
- 4dw - Delete 4 words
- cw - Change word
- x - Delete character at the cursor
- r - Replace character
- R - Overwrite characters from cursor onward
- s - Substitute one character under cursor continue to insert
- S - Substitute entire line and begin to insert at the beginning of the line
- k - Move cursor up
- j - Move cursor down
- h - Move cursor left
- l - Move cursor right
- :w - Save the file but keep it open
- :q - Quit without saving
- :wq - Save the file and quit

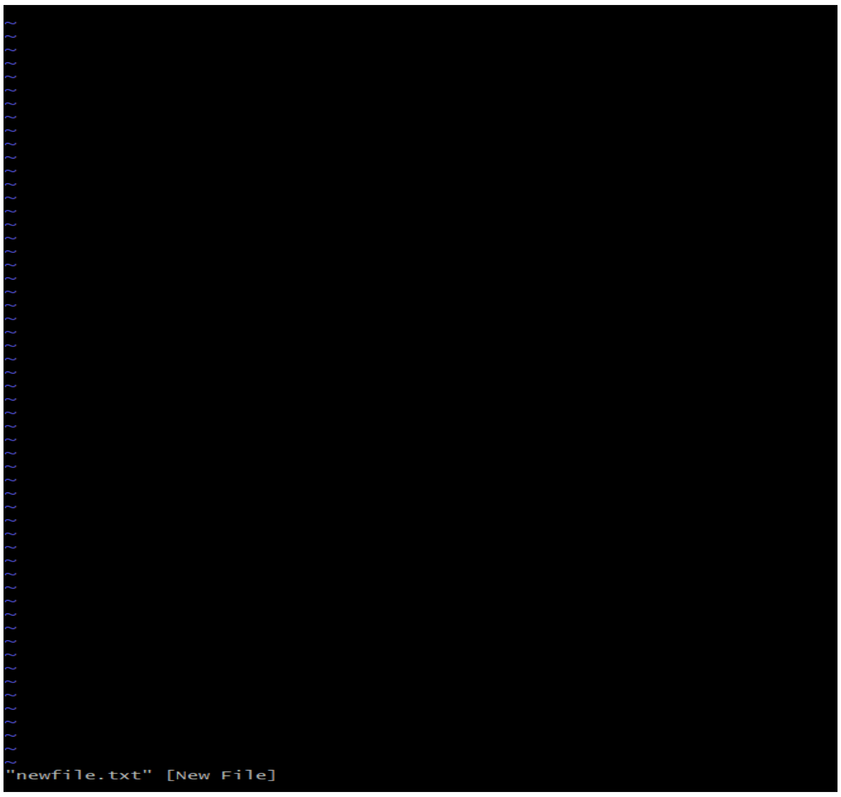
Note:

- Make sure you press the right command otherwise you will end up making undesirable changes to the file.
- You can also enter the insert mode by pressing a, A, o,

Opening of Vi Editor

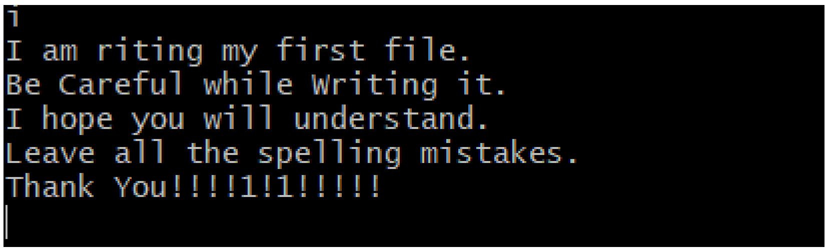


```
hp@DESKTOP-BQK27U3 ~
$ vi newfile.txt
```



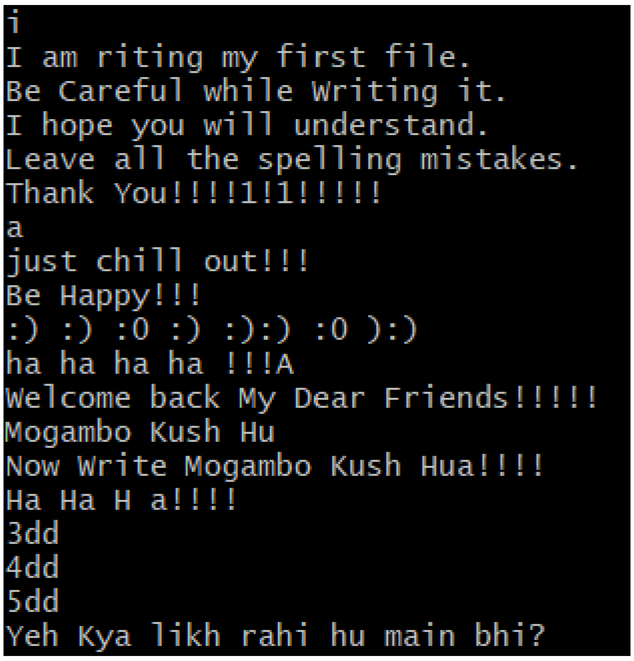
```
~
~
~
"newfile.txt" [New File]
```

Insert at cursor (goes into insert mode)



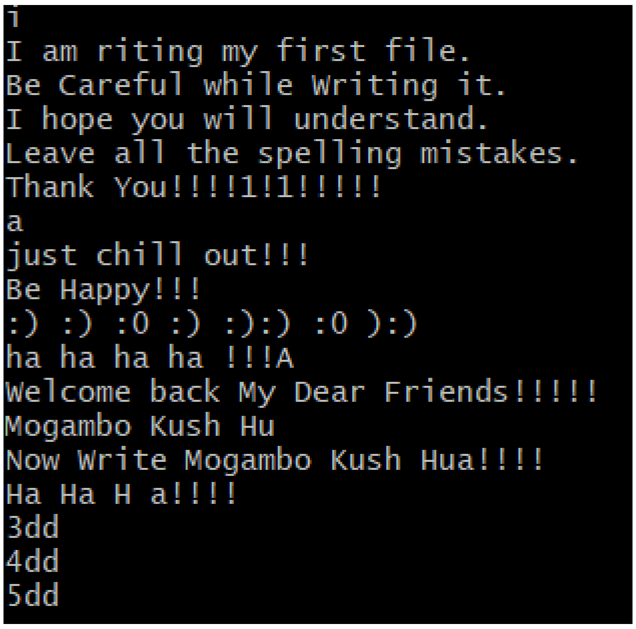
```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!
```

Write after cursor (goes into insert mode) and write at the end of line (goes into insert mode)



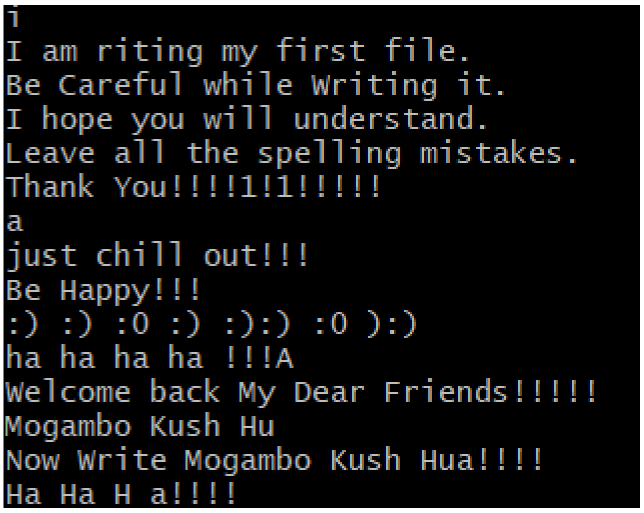
```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!
a
just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
Welcome back My Dear Friends!!!!!
Mogambo Kush Hu
Now Write Mogambo Kush Hua!!!!
Ha Ha H a!!!!
3dd
4dd
5dd
Yeh Kya likh rahi hu main bhi?
```

Delete line



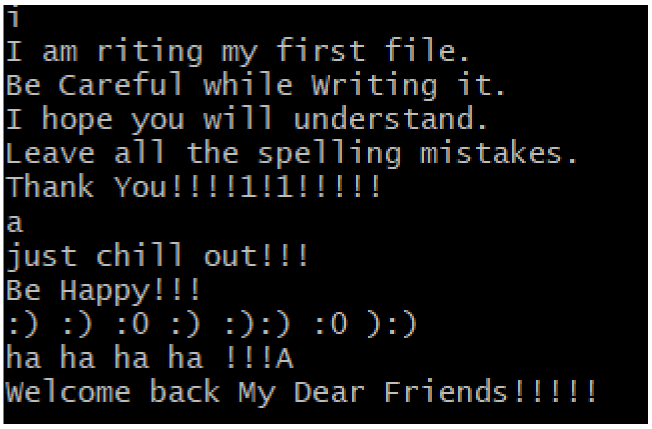
```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!
a
just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
Welcome back My Dear Friends!!!!!
Mogambo Kush Hu
Now Write Mogambo Kush Hua!!!!
Ha Ha H a!!!!
3dd
4dd
5dd
```

Delete 3 lines.

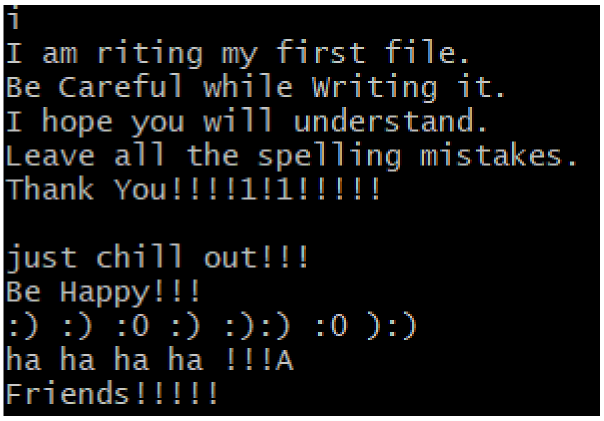


```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!
a
just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
Welcome back My Dear Friends!!!!!
Mogambo Kush Hu
Now Write Mogambo Kush Hua!!!!
Ha Ha H a!!!!
```

Delete 4 words

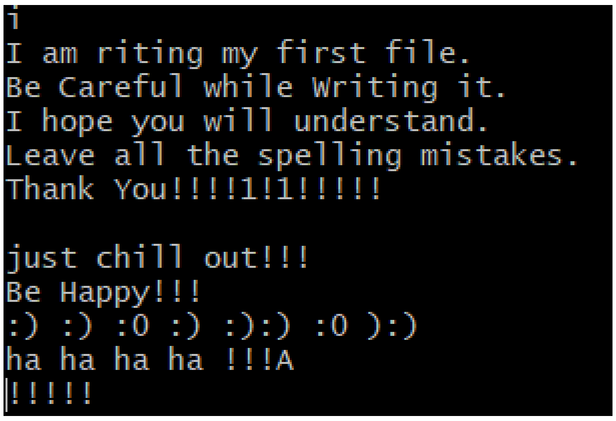


```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!
a
just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
Welcome back My Dear Friends!!!!!
```

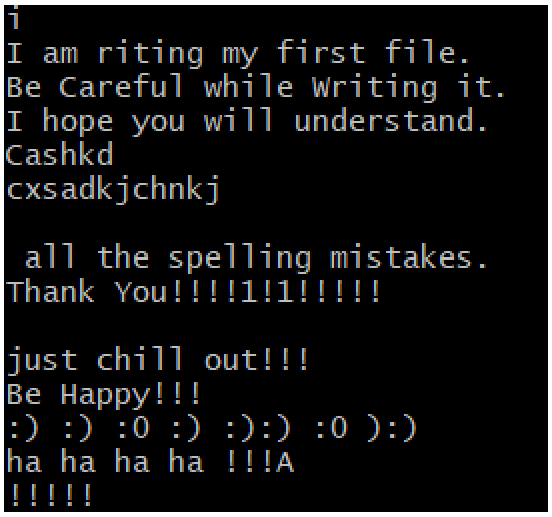


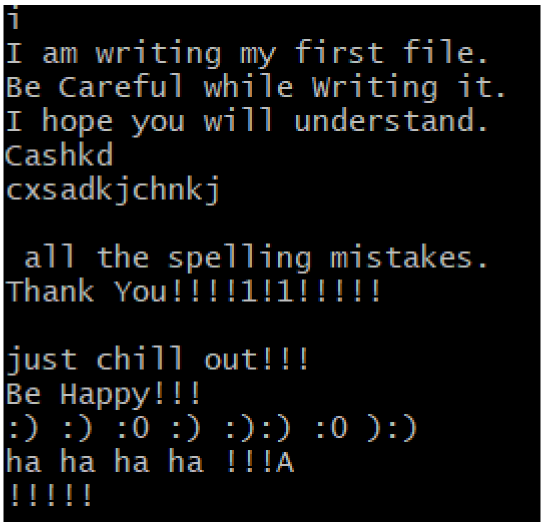
```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
Friends!!!!!
```

Delete word

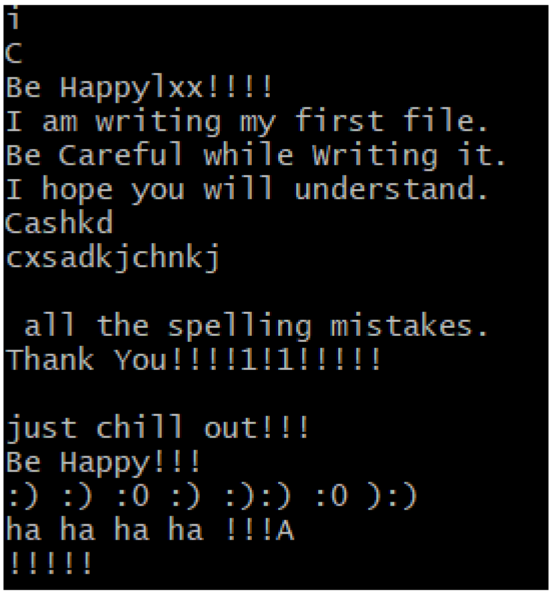

```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Leave all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
!!!!!
```

Change word

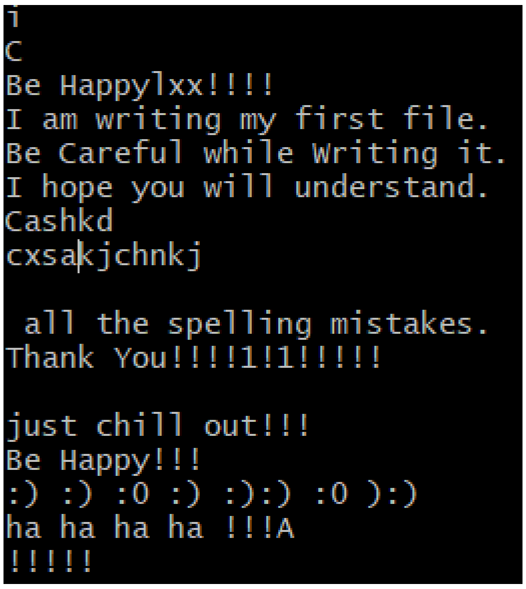

```
i
I am riting my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsadkjchnkj

 all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
!!!!!
```


```
i
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsadkjchnkj

 all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
!!!!!
```

Delete character at the cursor

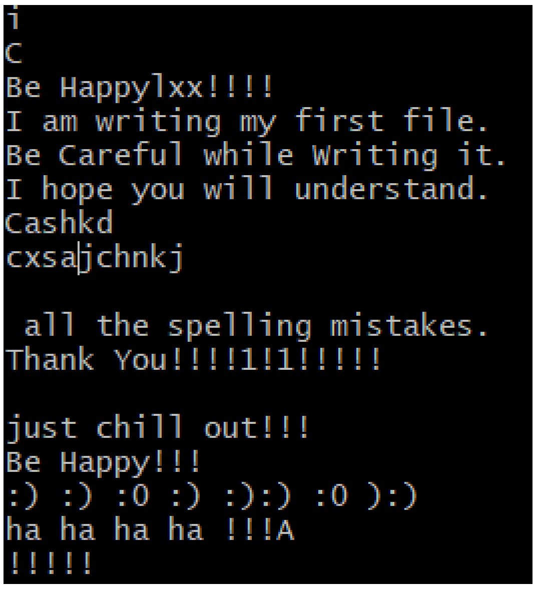


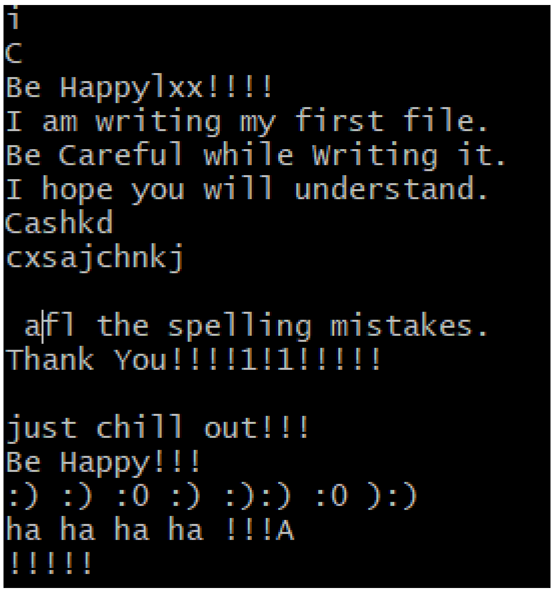
```
i
C
Be Happylxx!!!!
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsadkjchnkj

 all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
!!!!!
```



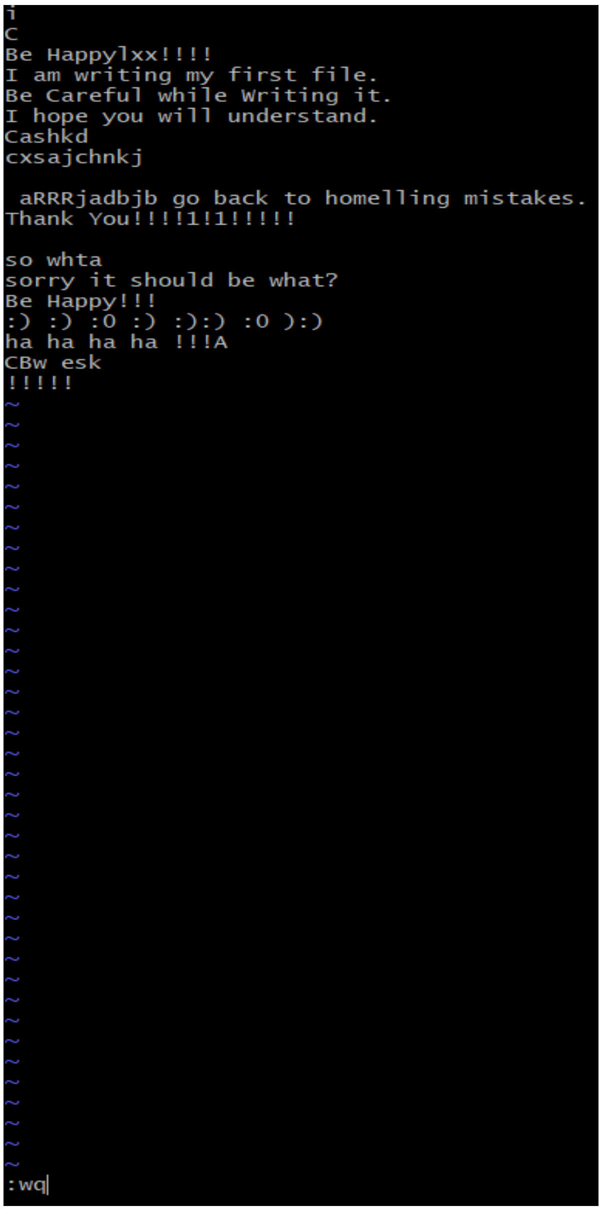
```
i
C
Be Happylxx!!!!
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsakjchnkj

 all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :0 :) :):) :0 ):)
ha ha ha ha !!!A
!!!!!
```

Replace character

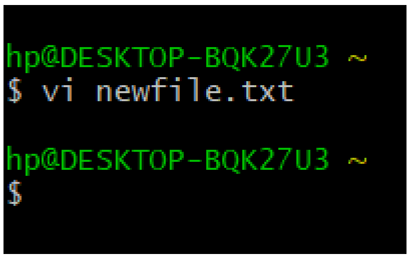

```
i
C
Be Happylxx!!!!
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsajchnkj

 all the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :O :) :):) :O ):)
ha ha ha ha !!!A
!!!!!
```


```
i
C
Be Happylxx!!!!
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsajchnkj

 afl the spelling mistakes.
Thank You!!!!1!1!!!!!

just chill out!!!
Be Happy!!!
:) :) :O :) :):) :O ):)
ha ha ha ha !!!A
!!!!!
```

Save the file and quit

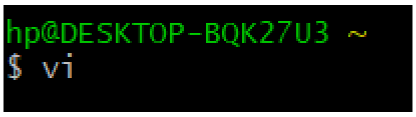


```
i
C
Be Happylxx!!!!
I am writing my first file.
Be Careful while Writing it.
I hope you will understand.
Cashkd
cxsajchnkj

 aRRRjadbjb go back to homelling mistakes.
Thank You!!!!1!1!!!!!

so whta
sorry it should be what?
Be Happy!!!
:) :) :O :) :):) :O ):)
ha ha ha ha !!!A
CBw esk
!!!!!
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:wq
```

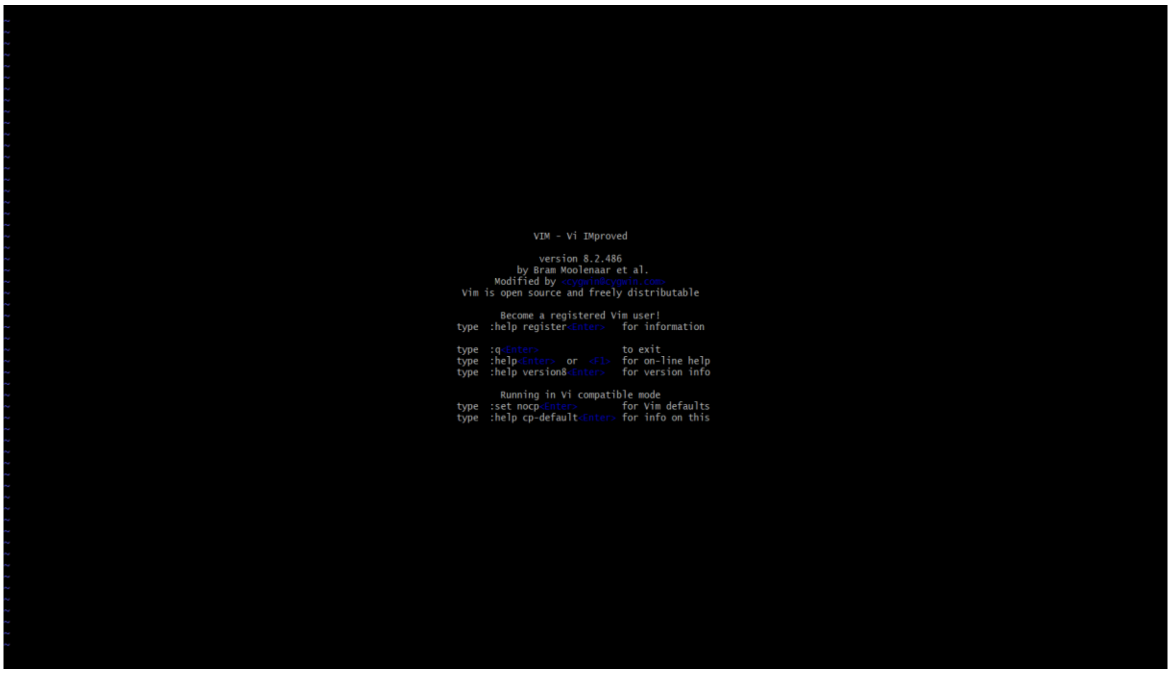


```
hp@DESKTOP-BQK27U3 ~
$ vi newfile.txt

hp@DESKTOP-BQK27U3 ~
$
```

Type “vi” at the prompt



```
hp@DESKTOP-BQK27U3 ~
$ vi
```



```
                 VIM - Vi IMproved

                  version 8.2.486
             by Bram Moolenaar et al.
        Modified by <cygwin@cygwin.com>
  Vim is open source and freely distributable

           Become a registered Vim user!
type  :help register<Enter>   for information

type  :q<Enter>               to exit
type  :help<Enter>  or  <F1>  for on-line help
type  :help version8<Enter>   for version info

           Running in Vi compatible mode
type  :set nocp<Enter>        for Vim defaults
type  :help cp-default<Enter> for info on this
```

<u>Results:</u> Linux commands has been executed successfully.

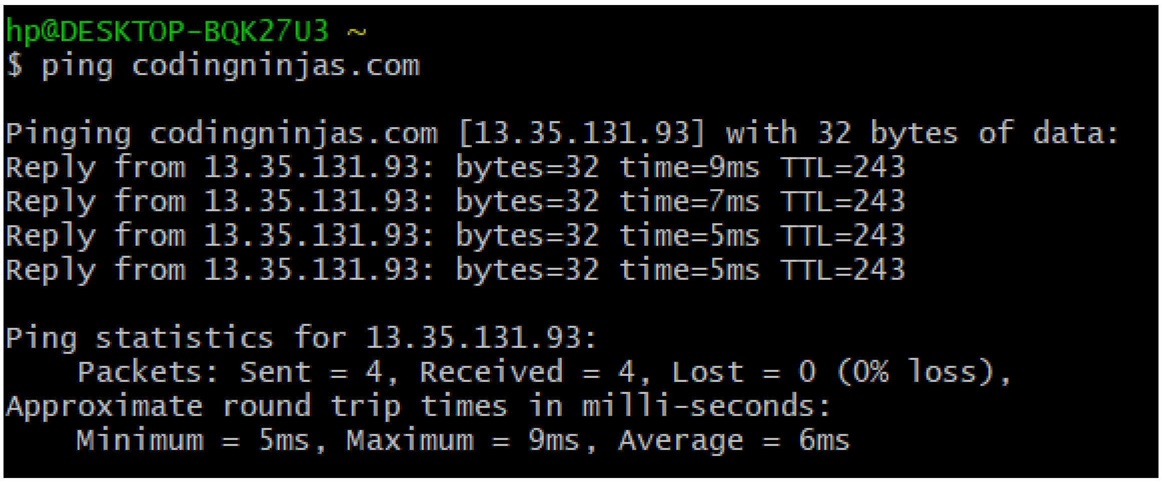
Experiment 4

Date: 14 – 01 - 2021

Aim: To Explore More Advanced Linux Commands.

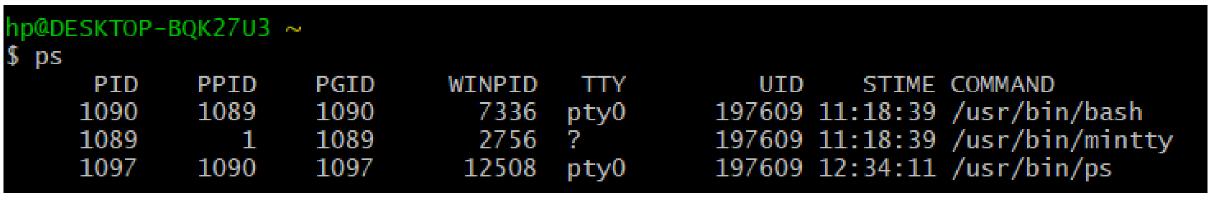
Theory:

1. **ping:** The ping command lets you verify that you have network connectivity with another network device. It is commonly used to help troubleshoot networking issues. To use ping, provide the IP address or machine name of the other device.
   Syntax: ping



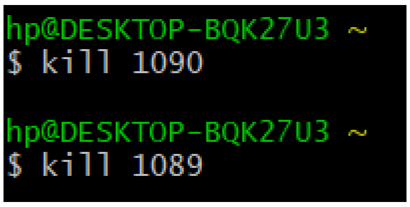
```
hp@DESKTOP-BQK27U3 ~
$ ping codingninjas.com

Pinging codingninjas.com [13.35.131.93] with 32 bytes of data:
Reply from 13.35.131.93: bytes=32 time=9ms TTL=243
Reply from 13.35.131.93: bytes=32 time=7ms TTL=243
Reply from 13.35.131.93: bytes=32 time=5ms TTL=243
Reply from 13.35.131.93: bytes=32 time=5ms TTL=243

Ping statistics for 13.35.131.93:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 5ms, Maximum = 9ms, Average = 6ms
```

2. **ps:** The ps command lists running processes. Using ps without any options causes it to list the processes running in the current shell.
   Syntax: ps



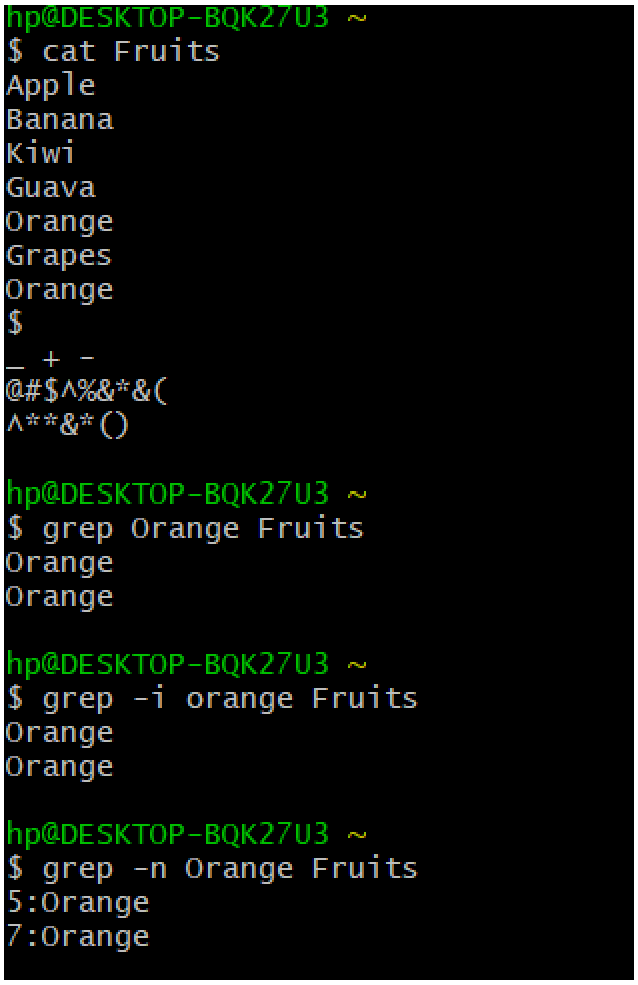
```
hp@DESKTOP-BQK27U3 ~
$ ps
      PID    PPID    PGID     WINPID   TTY         UID    STIME COMMAND
     1090    1089    1090       7336  pty0      197609 11:18:39 /usr/bin/bash
     1089       1    1089       2756  ?         197609 11:18:39 /usr/bin/mintty
     1097    1090    1097      12508  pty0      197609 12:34:11 /usr/bin/ps
```

3. **kill:**
   - To terminate a process use "kill"
   - Rules are simple:
     - ♦ You can kill all your own process.
     - ♦ Only root user can kill system level process.
     - ♦ Only root user can kill process started by other users.
   - used to terminate processes manually. *kill* command sends a signal to a process which terminates the process. If the user doesn't specify any signal which is to be sent along with kill command then default *TERM* signal is sent that terminates the process.
   - **Syntax:** kill <pid>

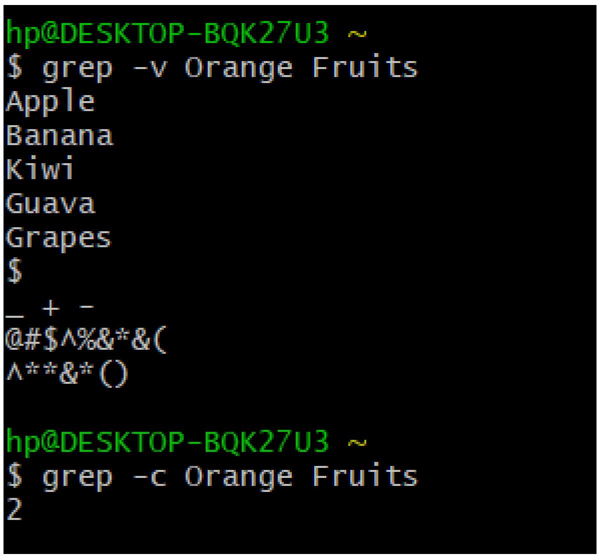


```
hp@DESKTOP-BQK27U3 ~
$ kill 1090

hp@DESKTOP-BQK27U3 ~
$ kill 1089
```

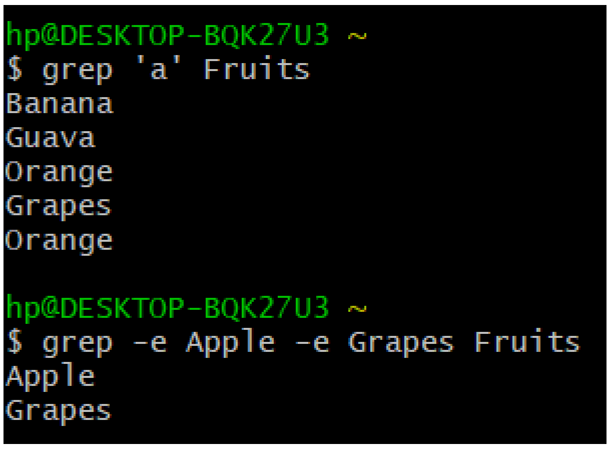
4. **grep:** It is used to grep searches the named input files for lines containing a match to the given pattern.
   Syntax: grep <options> <filename>
   Note: Options used in grep command are:
   - -e: pattern
   - -i: Ignore uppercase vs. lowercase.
   - -v: Invert match.
   - -c: Output count of matching lines only.
   - -l: Output matching files only.
   - -n: Precede each matching line with a line number.
   - -b: A historical curiosity: precede each matching line with a block number.
   - -h: Output matching lines without preceding them by file names.
   - -s: Suppress error messages about nonexistent or unreadable files.
   - -x
   - -f file: Take regexes from a file.
   - -o: Output the matched parts of a matching line.



```
hp@DESKTOP-BQK27U3 ~
$ cat Fruits
Apple
Banana
Kiwi
Guava
Orange
Grapes
Orange
$
_ + -
@#$^%&*&(
^**&*()

hp@DESKTOP-BQK27U3 ~
$ grep Orange Fruits
Orange
Orange

hp@DESKTOP-BQK27U3 ~
$ grep -i orange Fruits
Orange
Orange

hp@DESKTOP-BQK27U3 ~
$ grep -n Orange Fruits
5:Orange
7:Orange
```

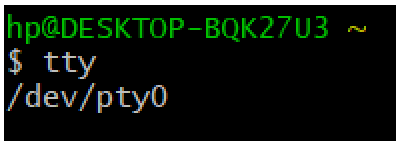


```
hp@DESKTOP-BQK27U3 ~
$ grep -v Orange Fruits
Apple
Banana
Kiwi
Guava
Grapes
$
_ + -
@#$^%&*&(
^**&*()

hp@DESKTOP-BQK27U3 ~
$ grep -c Orange Fruits
2
```



```
hp@DESKTOP-BQK27U3 ~
$ grep 'a' Fruits
Banana
Guava
Orange
Grapes
Orange

hp@DESKTOP-BQK27U3 ~
$ grep -e Apple -e Grapes Fruits
Apple
Grapes
```

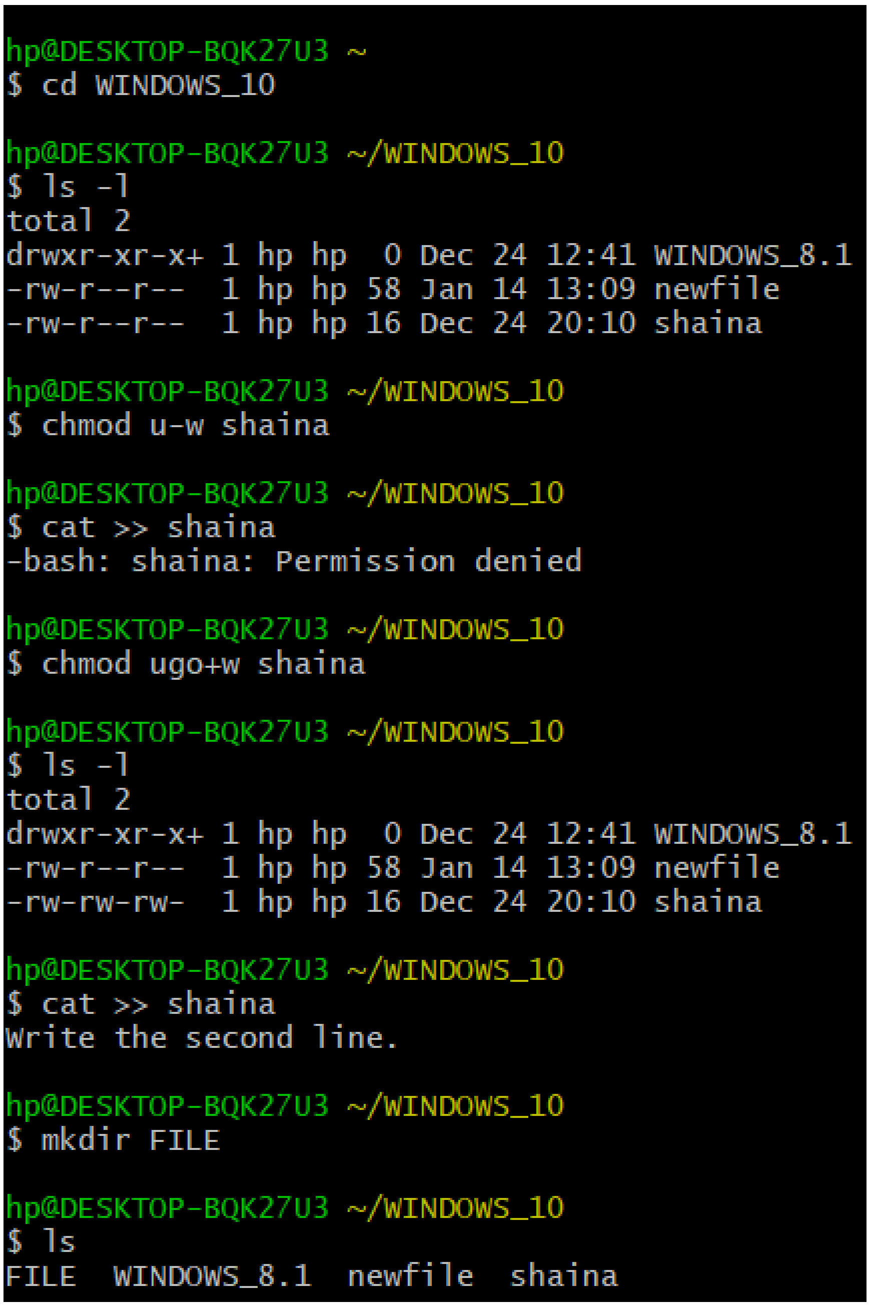
5. **tty:**
   - tty is a command in Unix and Unix-like operating systems to print the file name of the terminal connected to standard input. tty stands for TeleTYpewriter.
   - The tty command basically prints the file name of the terminal connected to standard input.
   - **Syntax:** tty



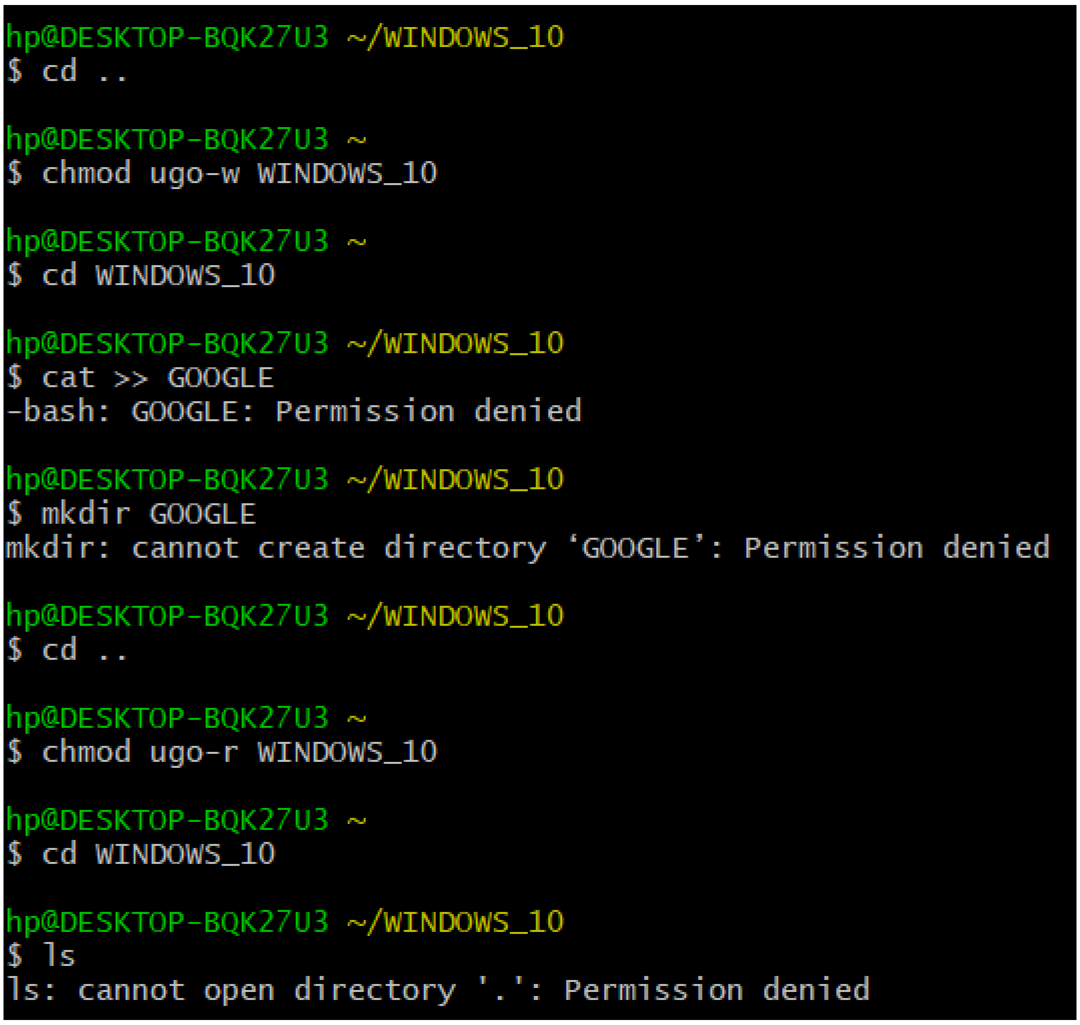
```
hp@DESKTOP-BQK27U3 ~
$ tty
/dev/pty0
```

6. **chmod:**
   - There are three types of permissions: read (r), write (w), and execute (x).
   - To read a file is to view its contents. For example, a text file must have read permission for someone to read the text within.
   - If the user wants to add a sentence to that file, it needs write permission.
   - The execute permission enables someone to run a file, such as a shell script or a binary program file.
   - The ls -l command displays the permissions assigned to a file.

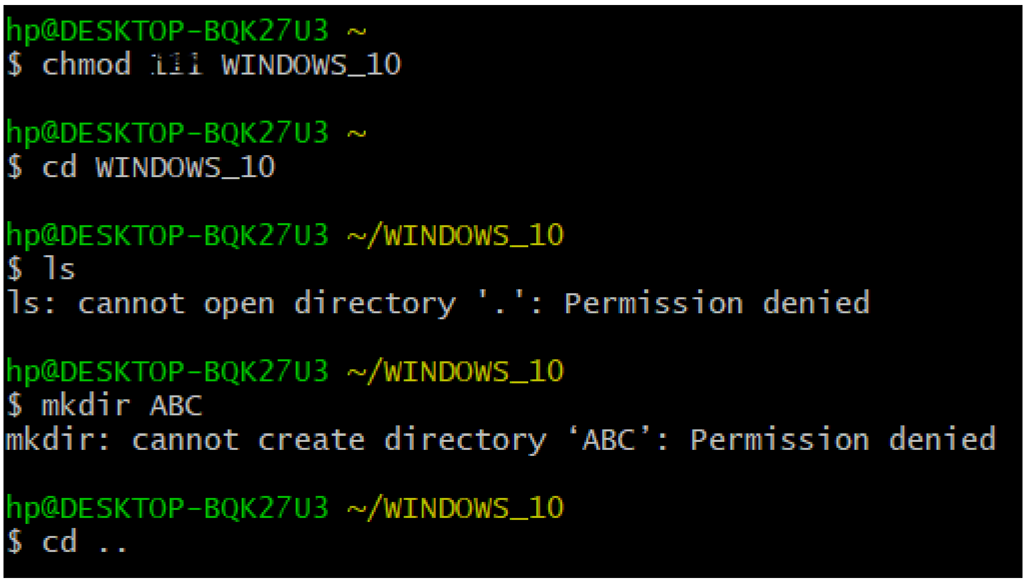
- *user, group,* and *other*.
- Each file is associated with an owner and a group and assigned with permission access rights for three different classes of users:
- The file owner.
- The group members.
- Others (everybody else)
- **Operation:**
  - - Removes the specified permissions.
  - + Adds specified permissions.
  - = Changes the current permissions to the specified permissions. If no permissions are specified after the = symbol, all permissions from the specified user class are removed.
- Directories are special types of files that contain other files and directories.
- The chmod command allows you to change the permissions on a file using either a symbolic or numeric mode or a reference file.
- **Symbolic (Text) Method:**
  - **Syntax:** chmod <options> <ugoa…><-+=perm s> <file>
  - The permissions (perms...) can be explicitly set using either zero or one or more of the following letters: r, w, x. Use a single letter from the set u, g, and o when copying permissions from one to another user class.
- **Numeric Method:**
  - The syntax of the chmod command when using numeric method has the following format.
  - **Syntax:** chmod <options> <Number File...>
  - When using the numeric mode, you can set the permissions for all three user classes (owner, group, and all others) at the same time. the first digit represents the permissions of the file's owner, the second one the file's group, and the last one all other users.
  - Each write, read, and execute permissions have the following number value:
    - r (read) = 4
    - w (write) = 2
    - x (execute) = 1
    - no permissions = 0
  - The permissions number of a specific user class is represented by the sum of the values of the permissions for that group.
  - calculate the totals for all users classes. For example, to give read, write and execute permission to the file's owner, read and execute permissions to the file's group and only read permissions to all other users you would do the following:
    - Owner: rwx=4+2+1=7
    - Group: r-x=4+0+1=5
    - Others: r-x=4+0+0=4
  - Using the method above we come up to the number 754, which represents the desired permissions.



```
hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls -l
total 2
drwxr-xr-x+ 1 hp hp  0 Dec 24 12:41 WINDOWS_8.1
-rw-r--r--  1 hp hp 58 Jan 14 13:09 newfile
-rw-r--r--  1 hp hp 16 Dec 24 20:10 shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ chmod u-w shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cat >> shaina
-bash: shaina: Permission denied

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ chmod ugo+w shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls -l
total 2
drwxr-xr-x+ 1 hp hp  0 Dec 24 12:41 WINDOWS_8.1
-rw-r--r--  1 hp hp 58 Jan 14 13:09 newfile
-rw-rw-rw-  1 hp hp 16 Dec 24 20:10 shaina

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cat >> shaina
Write the second line.

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ mkdir FILE

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
FILE  WINDOWS_8.1  newfile  shaina
```



```
hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd ..

hp@DESKTOP-BQK27U3 ~
$ chmod ugo-w WINDOWS_10

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cat >> GOOGLE
-bash: GOOGLE: Permission denied

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ mkdir GOOGLE
mkdir: cannot create directory ‘GOOGLE’: Permission denied

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd ..

hp@DESKTOP-BQK27U3 ~
$ chmod ugo-r WINDOWS_10

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
ls: cannot open directory '.': Permission denied
```

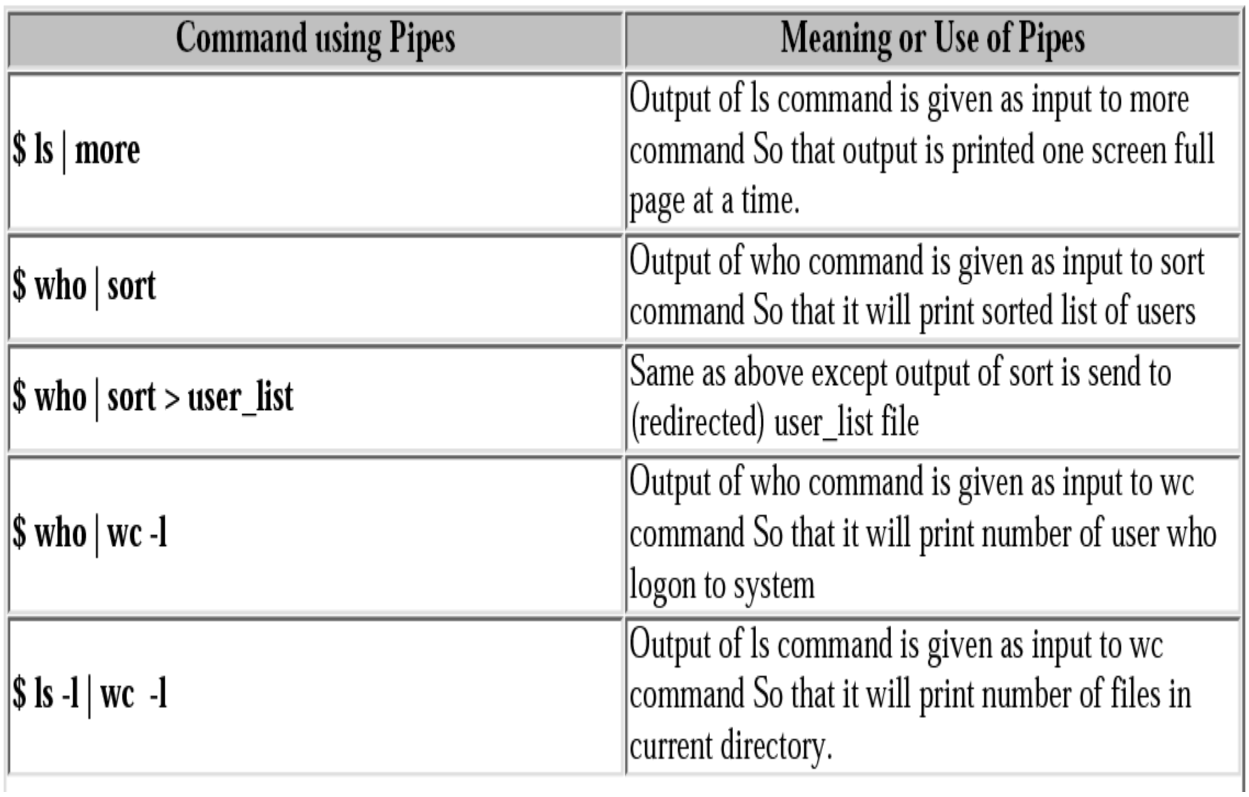


```
hp@DESKTOP-BQK27U3 ~
$ chmod 111 WINDOWS_10

hp@DESKTOP-BQK27U3 ~
$ cd WINDOWS_10

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ ls
ls: cannot open directory '.': Permission denied

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ mkdir ABC
mkdir: cannot create directory ‘ABC’: Permission denied

hp@DESKTOP-BQK27U3 ~/WINDOWS_10
$ cd ..
```

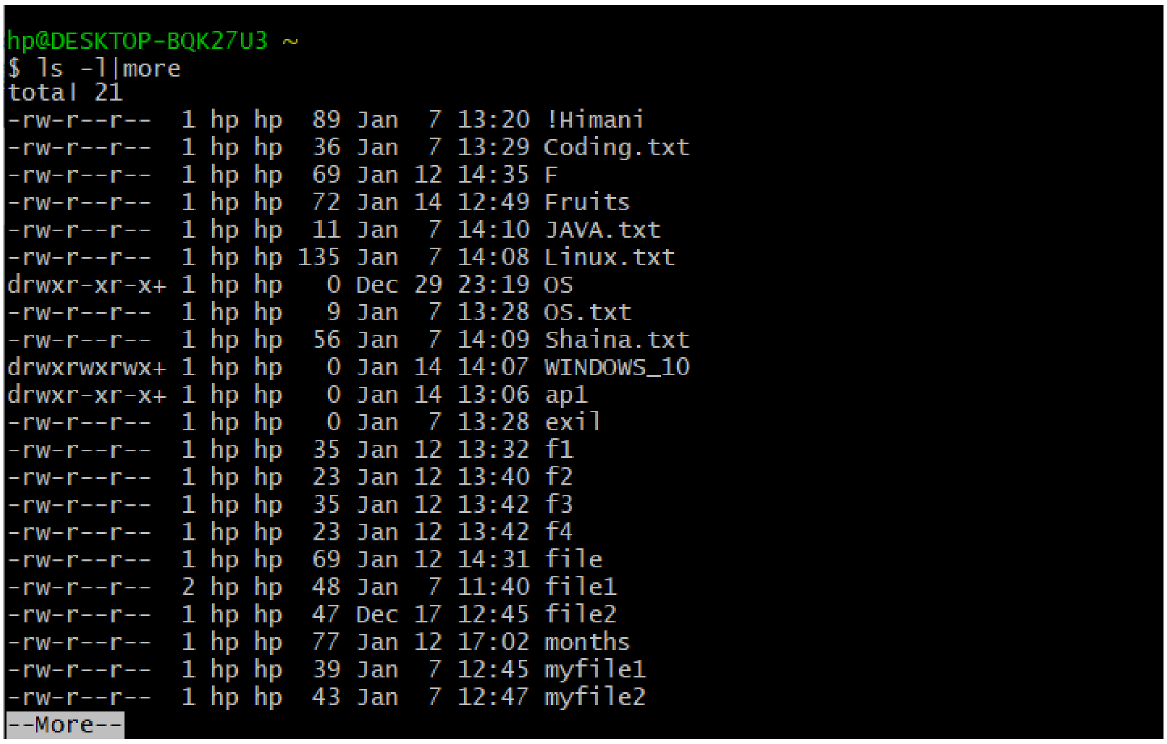
7. **Pipe (|):**

- A pipe is a form of redirection (transfer of standard output to some other destination) that is used in Linux and other Unix-like operating systems to send the output of one command/program/process to another command/program/process for further processing.

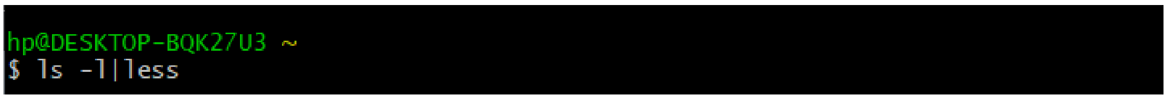
- You can make it do so by using the pipe character ‘|’.
- It can also be visualized as a temporary connection between two or more commands/ programs/ processes.
- Pipes are unidirectional i.e. data flows from left to right through the pipeline.
- The pipe acts as a container which takes the output of ls -l and gives it to more as input. This command does not use a disk to connect standard output of ls -l to the standard input of more because pipe is implemented in the main memory.
- **Syntax:** command 1| command 2| command 3|……. | command n



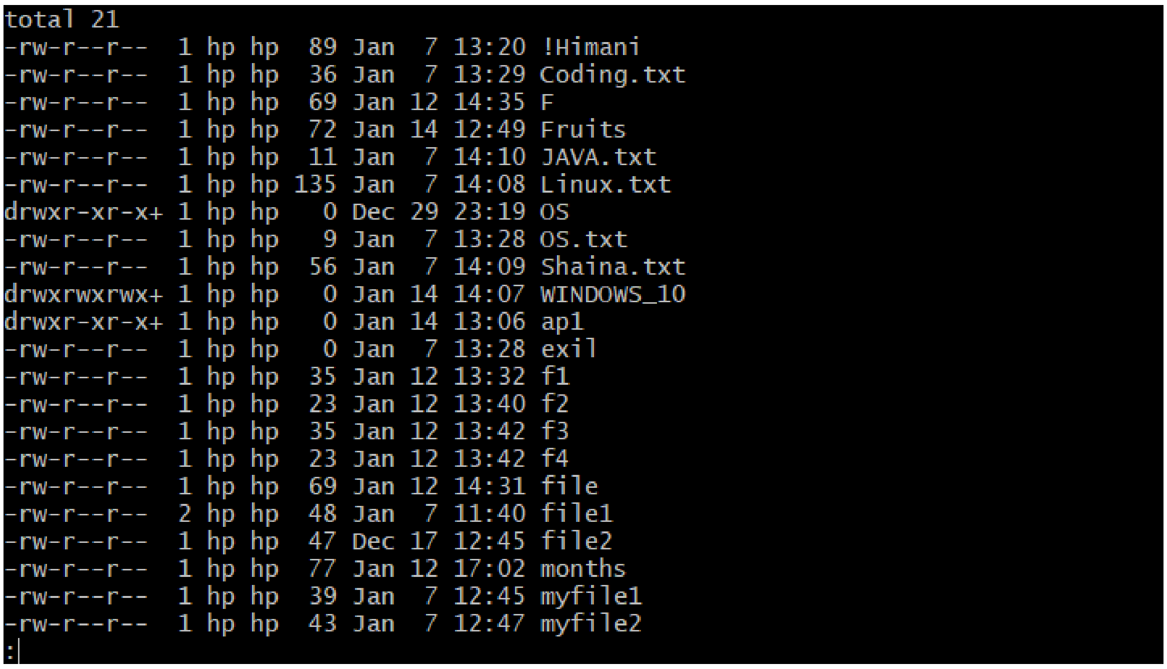
Command using Pipes	Meaning or Use of Pipes
$ ls | more	Output of ls command is given as input to more command So that output is printed one screen full page at a time.
$ who | sort	Output of who command is given as input to sort command So that it will print sorted list of users
$ who | sort > user_list	Same as above except output of sort is send to (redirected) user_list file
$ who | wc -l	Output of who command is given as input to wc command So that it will print number of user who logon to system
$ ls -l | wc -l	Output of ls command is given as input to wc command So that it will print number of files in current directory.



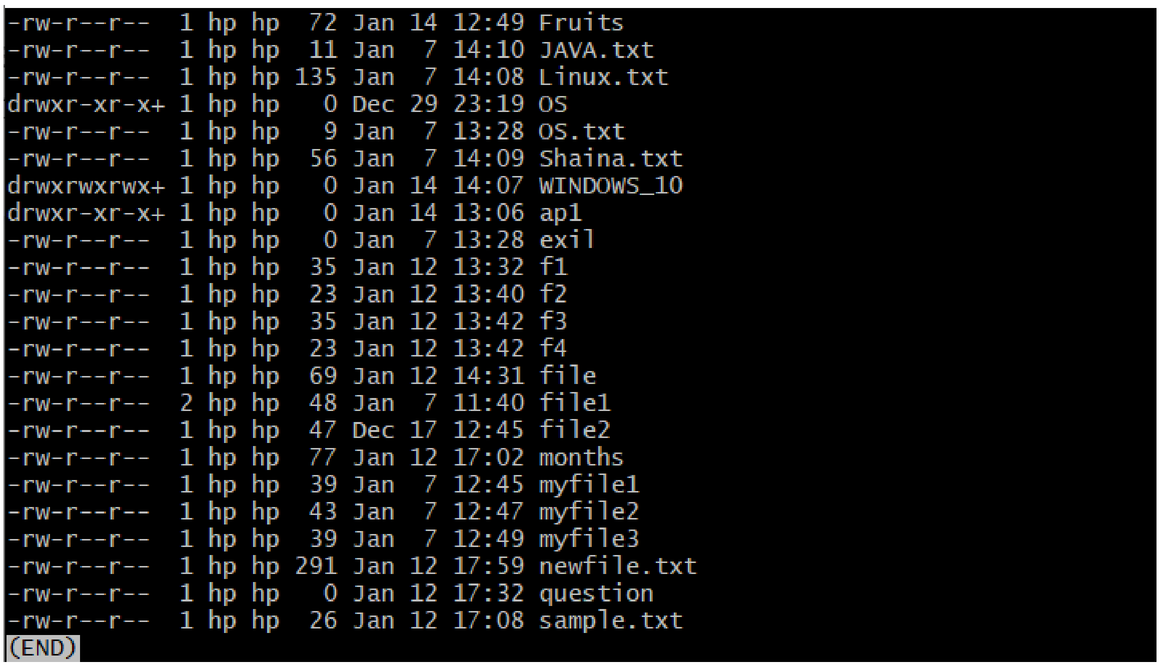
```
hp@DESKTOP-BQK27U3 ~
$ ls -l|more
total 21
-rw-r--r--  1 hp hp  89 Jan  7 13:20 !Himani
-rw-r--r--  1 hp hp  36 Jan  7 13:29 Coding.txt
-rw-r--r--  1 hp hp  69 Jan 12 14:35 F
-rw-r--r--  1 hp hp  72 Jan 14 12:49 Fruits
-rw-r--r--  1 hp hp  11 Jan  7 14:10 JAVA.txt
-rw-r--r--  1 hp hp 135 Jan  7 14:08 Linux.txt
drwxr-xr-x+ 1 hp hp   0 Dec 29 23:19 OS
-rw-r--r--  1 hp hp   9 Jan  7 13:28 OS.txt
-rw-r--r--  1 hp hp  56 Jan  7 14:09 Shaina.txt
drwxrwxrwx+ 1 hp hp   0 Jan 14 14:07 WINDOWS_10
drwxr-xr-x+ 1 hp hp   0 Jan 14 13:06 ap1
-rw-r--r--  1 hp hp   0 Jan  7 13:28 exil
-rw-r--r--  1 hp hp  35 Jan 12 13:32 f1
-rw-r--r--  1 hp hp  23 Jan 12 13:40 f2
-rw-r--r--  1 hp hp  35 Jan 12 13:42 f3
-rw-r--r--  1 hp hp  23 Jan 12 13:42 f4
-rw-r--r--  1 hp hp  69 Jan 12 14:31 file
-rw-r--r--  2 hp hp  48 Jan  7 11:40 file1
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
-rw-r--r--  1 hp hp  77 Jan 12 17:02 months
-rw-r--r--  1 hp hp  39 Jan  7 12:45 myfile1
-rw-r--r--  1 hp hp  43 Jan  7 12:47 myfile2
--More--
```



```
hp@DESKTOP-BQK27U3 ~
$ ls -l|less
```



```
total 21
-rw-r--r--  1 hp hp  89 Jan  7 13:20 !Himani
-rw-r--r--  1 hp hp  36 Jan  7 13:29 Coding.txt
-rw-r--r--  1 hp hp  69 Jan 12 14:35 F
-rw-r--r--  1 hp hp  72 Jan 14 12:49 Fruits
-rw-r--r--  1 hp hp  11 Jan  7 14:10 JAVA.txt
-rw-r--r--  1 hp hp 135 Jan  7 14:08 Linux.txt
drwxr-xr-x+ 1 hp hp   0 Dec 29 23:19 OS
-rw-r--r--  1 hp hp   9 Jan  7 13:28 OS.txt
-rw-r--r--  1 hp hp  56 Jan  7 14:09 Shaina.txt
drwxrwxrwx+ 1 hp hp   0 Jan 14 14:07 WINDOWS_10
drwxr-xr-x+ 1 hp hp   0 Jan 14 13:06 ap1
-rw-r--r--  1 hp hp   0 Jan  7 13:28 exil
-rw-r--r--  1 hp hp  35 Jan 12 13:32 f1
-rw-r--r--  1 hp hp  23 Jan 12 13:40 f2
-rw-r--r--  1 hp hp  35 Jan 12 13:42 f3
-rw-r--r--  1 hp hp  23 Jan 12 13:42 f4
-rw-r--r--  1 hp hp  69 Jan 12 14:31 file
-rw-r--r--  2 hp hp  48 Jan  7 11:40 file1
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
-rw-r--r--  1 hp hp  77 Jan 12 17:02 months
-rw-r--r--  1 hp hp  39 Jan  7 12:45 myfile1
-rw-r--r--  1 hp hp  43 Jan  7 12:47 myfile2
:
```


```
-rw-r--r--  1 hp hp  72 Jan 14 12:49 Fruits
-rw-r--r--  1 hp hp  11 Jan  7 14:10 JAVA.txt
-rw-r--r--  1 hp hp 135 Jan  7 14:08 Linux.txt
drwxr-xr-x+ 1 hp hp   0 Dec 29 23:19 OS
-rw-r--r--  1 hp hp   9 Jan  7 13:28 OS.txt
-rw-r--r--  1 hp hp  56 Jan  7 14:09 Shaina.txt
drwxrwxrwx+ 1 hp hp   0 Jan 14 14:07 WINDOWS_10
drwxr-xr-x+ 1 hp hp   0 Jan 14 13:06 ap1
-rw-r--r--  1 hp hp   0 Jan  7 13:28 exil
-rw-r--r--  1 hp hp  35 Jan 12 13:32 f1
-rw-r--r--  1 hp hp  23 Jan 12 13:40 f2
-rw-r--r--  1 hp hp  35 Jan 12 13:42 f3
-rw-r--r--  1 hp hp  23 Jan 12 13:42 f4
-rw-r--r--  1 hp hp  69 Jan 12 14:31 file
-rw-r--r--  2 hp hp  48 Jan  7 11:40 file1
-rw-r--r--  1 hp hp  47 Dec 17 12:45 file2
-rw-r--r--  1 hp hp  77 Jan 12 17:02 months
-rw-r--r--  1 hp hp  39 Jan  7 12:45 myfile1
-rw-r--r--  1 hp hp  43 Jan  7 12:47 myfile2
-rw-r--r--  1 hp hp  39 Jan  7 12:49 myfile3
-rw-r--r--  1 hp hp 291 Jan 12 17:59 newfile.txt
-rw-r--r--  1 hp hp   0 Jan 12 17:32 question
-rw-r--r--  1 hp hp  26 Jan 12 17:08 sample.txt
(END)
```

Note:

1. **more:** more command is used to view the text files in the command prompt, displaying one screen at a time in case the file is large (For example log files). The more command also allows the user do scroll up and down through the page. The syntax along with options and command is as follows. Another application of more is to use it with some other command after a pipe. When the output is large, we can use more command to see output one by one.
   Syntax: more <-options> <-num] ><+/pattern> <+linenum> <file_name>
   - **[-options]:** any option that you want to use in order to change the way the file is displayed. Choose any one from the followings: (-d, -l, -f, -p, -c, -s, -u)
   - **[-num]:** type the number of lines that you want to display per screen.
   - **[+/pattern]:** replace the pattern with any string that you want to find in the text file.
   - **[+linenum]:** use the line number from where you want to start displaying the text content.
   - **[file_name]:** name of the file containing the text that you want to display on the screen.
2. **less:** Less command is linux utility which can be used to read contents of text file one page(one screen) per time. It has faster access because if file is large, it don't access complete file, but access it page by page. For example, if it's a large file and you are reading it using any text editor, then the complete file will be loaded to main memory, but less command don't load entire file, but load it part by part, which makes it faster.
   Syntax: less <filename>
   Options used in less command are:
   - -E : causes less to automatically exit the first time it reaches end of file.
   - -f : forces non-regular file to open.
   - -F : causes less to exit if entire file can be displayed on first screen
   - -g : highlight the string which was found by last search command
   - -G : suppresses all highlighting of strings found by search commands

- -i : cause searches to ignore case
- -n : suppresses line numbers
- -p pattern : it tells less to start at the first occurrence of pattern in the file
- -s : causes consecutive blank lines to be squeezed into a single blank line

Result: More linux commands has been explored successfully.

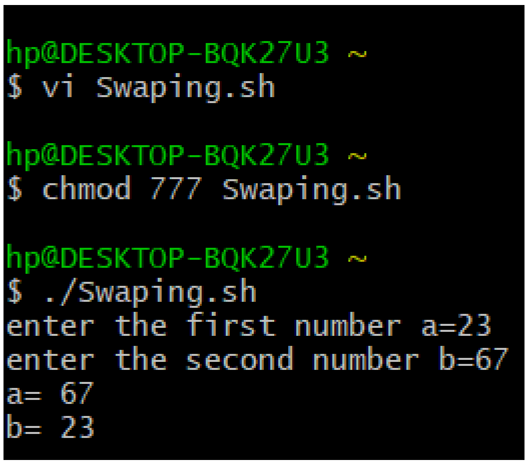
Experiment 5

Q1) Write a program to swap 2 numbers.

Code:

```
#!/bin/bash
read -p "enter the first number a=" a
read -p "enter the second number b=" b
temp=`expr $a`
a=`expr $b`
b=`expr $temp`
echo "a=" $a
echo "b=" $b
```

Output:



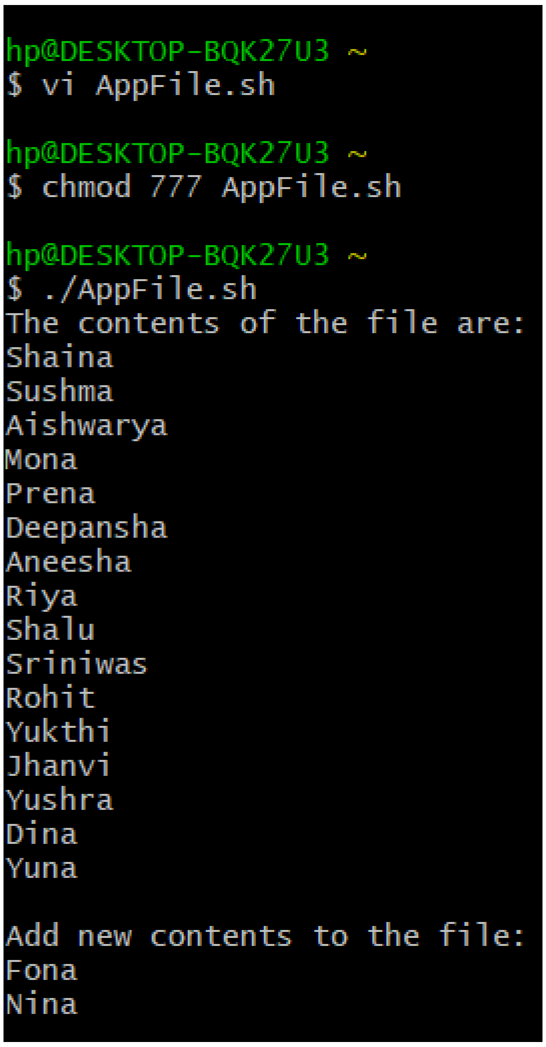
```
hp@DESKTOP-BQK27U3 ~
$ vi Swaping.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 Swaping.sh

hp@DESKTOP-BQK27U3 ~
$ ./Swaping.sh
enter the first number a=23
enter the second number b=67
a= 67
b= 23
```

Q2) Write a script to add some text in already existing file.

Code:

```
#!/bin/bash
echo "The contents of the file are:"
cat MyFile
echo " "
echo "Add new contents to the file:"
cat >> MyFile
```

Output:



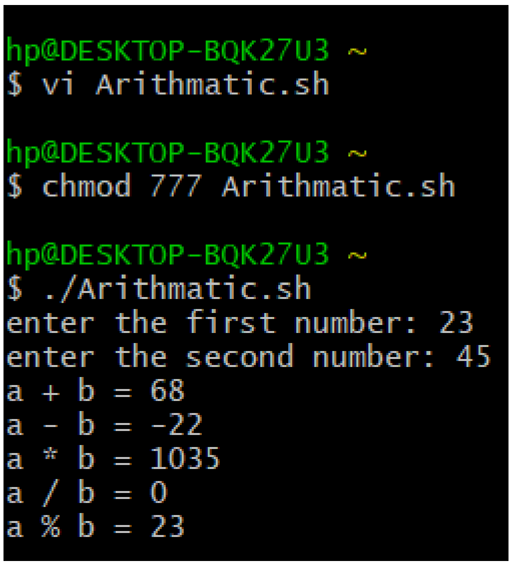
```
hp@DESKTOP-BQK27U3 ~
$ vi AppFile.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 AppFile.sh

hp@DESKTOP-BQK27U3 ~
$ ./AppFile.sh
The contents of the file are:
Shaina
Sushma
Aishwarya
Mona
Prena
Deepansha
Aneesha
Riya
Shalu
Sriniwas
Rohit
Yukthi
Jhanvi
Yushra
Dina
Yuna

Add new contents to the file:
Fona
Nina
```

Q3) Write a script to demonstrate use of arithmetic operator.

Code:

```
#!/bin/bash

read -p "enter the first number: " a;

read -p "enter the second number: " b;

c=`expr $a + $b`

d=`expr $a - $b`

e=`expr $a "*" $b`

f=`expr $a / $b`

g=`expr $a % $b`

echo "a + b =" $c

echo "a - b =" $d
echo "a "*" b =" $e

echo "a / b =" $f

echo "a % b =" $g
```

Output:



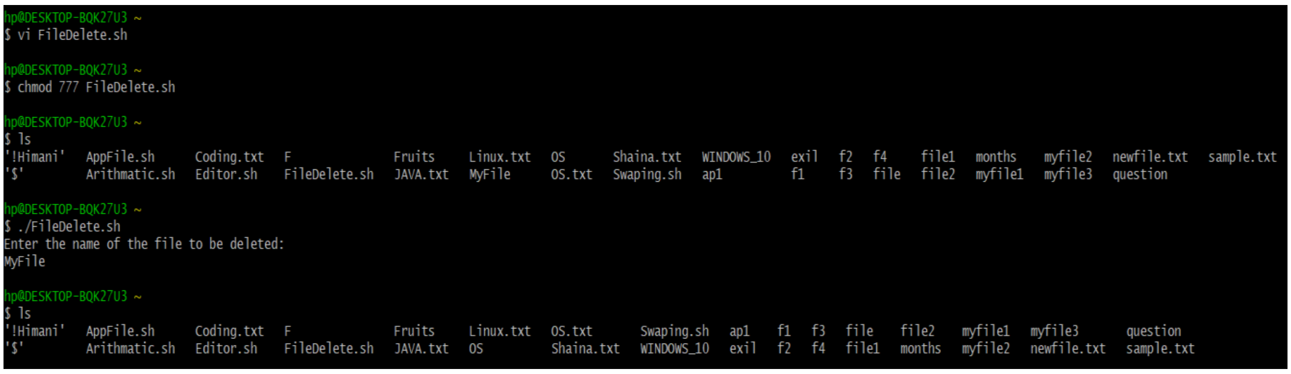
```
hp@DESKTOP-BQK27U3 ~
$ vi Arithmatic.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 Arithmatic.sh

hp@DESKTOP-BQK27U3 ~
$ ./Arithmatic.sh
enter the first number: 23
enter the second number: 45
a + b = 68
a - b = -22
a * b = 1035
a / b = 0
a % b = 23
```

Q4) Write a script to delete file.

Code:

```
#!/bin/bash

echo "Enter the name of the file to be deleted:"

read fname

rm $fname
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi FileDelete.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 FileDelete.sh

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'  AppFile.sh     Coding.txt  F              Fruits    Linux.txt  OS      Shaina.txt  WINDOWS_10  exil  f2  f4    file1  months   myfile2  newfile.txt  sample.txt
'$'        Arithmatic.sh  Editor.sh   FileDelete.sh  JAVA.txt  MyFile     OS.txt  Swaping.sh  ap1         f1    f3  file  file2  myfile1  myfile3  question

hp@DESKTOP-BQK27U3 ~
$ ./FileDelete.sh
Enter the name of the file to be deleted:
MyFile

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'  AppFile.sh     Coding.txt  F              Fruits    Linux.txt  OS.txt      Swaping.sh  ap1   f1  f3  file   file2   myfile1  myfile3      question
'$'        Arithmatic.sh  Editor.sh   FileDelete.sh  JAVA.txt  OS         Shaina.txt  WINDOWS_10  exil  f2  f4  file1  months  myfile2  newfile.txt  sample.txt
```

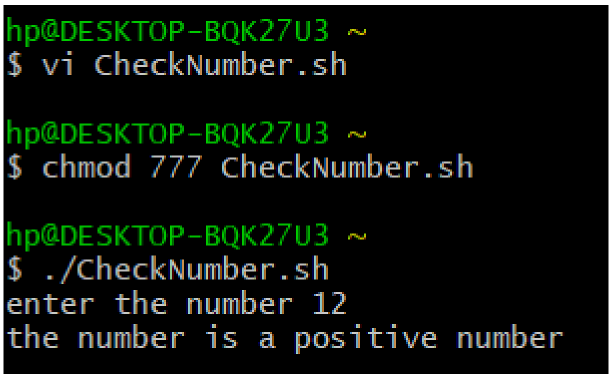
Experiment 6

Q1) Write a script to check whether number is positive or negative.

Code:

```
#!/bin/bash

read -p "enter the number" a

if [ $a -gt 0 ]

then

echo "the number is a positive number"

elif [ $a -le 0 ]

then

echo "the number is a negative number"

else

echo "Zero"

fi
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi CheckNumber.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 CheckNumber.sh

hp@DESKTOP-BQK27U3 ~
$ ./CheckNumber.sh
enter the number 12
the number is a positive number
```

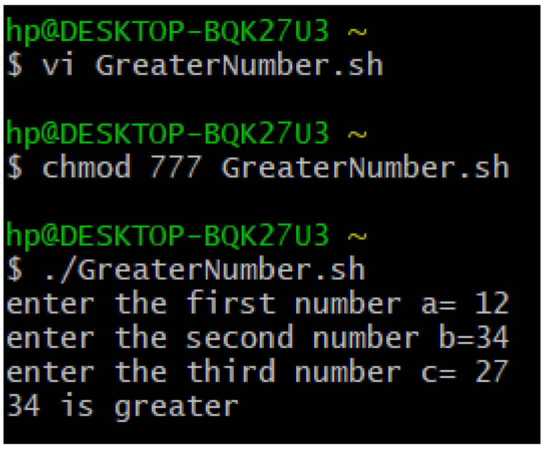
Q2) Write a script to find greatest number among three numbers.

Code:

```
#!/bin/bash

read -p "enter the first number a=" a

read -p "enter the second number b=" b

read -p "enter the third number c=" c

if [ $a -gt $b -a $a -gt $c ]

then

echo "$a is greater"
```

```
elif [ $b -gt $a -a $b -gt $c ]

then

echo "$b is greater"

else

echo "$c is greater"

fi
```

Output:



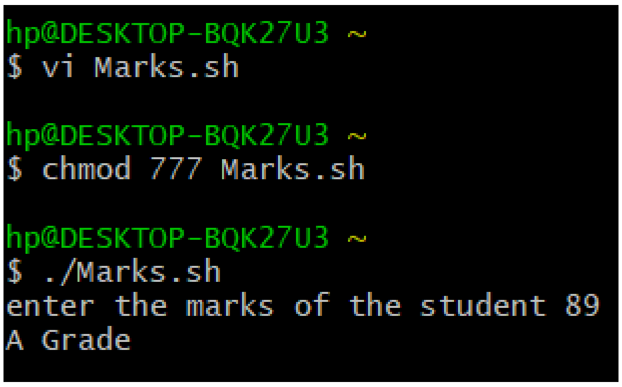
```
hp@DESKTOP-BQK27U3 ~
$ vi GreaterNumber.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 GreaterNumber.sh

hp@DESKTOP-BQK27U3 ~
$ ./GreaterNumber.sh
enter the first number a= 12
enter the second number b=34
enter the third number c= 27
34 is greater
```

Q3) Write a script to enter the marks of a student. If the marks are greater than 70 display grade A, if the grade is greater than 60 and less than 70 display grade B, else display "Fail".

Code:

```
#!/bin/bash

read -p "enter the marks of the student" m

if [ $m -gt 70 ]

then

echo "A Grade"

elif [ $m -le 70 -a $m -gt 60 ]

then

echo "B Grade"

else

echo "Fail"

fi
```

Output:



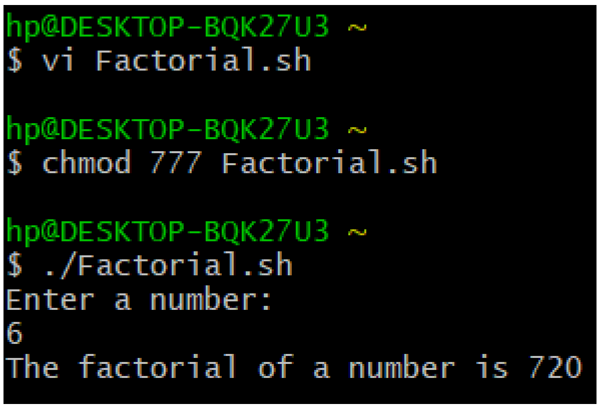
```
hp@DESKTOP-BQK27U3 ~
$ vi Marks.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 Marks.sh

hp@DESKTOP-BQK27U3 ~
$ ./Marks.sh
enter the marks of the student 89
A Grade
```

Q4) Write a script to calculate factorial of a number.

Code:

```
#!/bin/bash
echo Enter a number:
read a
fact=1
for (( i=$a; i > 1; i-- ))
{
fact=$((fact*i))
}
echo The factorial of a number is $fact
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi Factorial.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 Factorial.sh

hp@DESKTOP-BQK27U3 ~
$ ./Factorial.sh
Enter a number:
6
The factorial of a number is 720
```

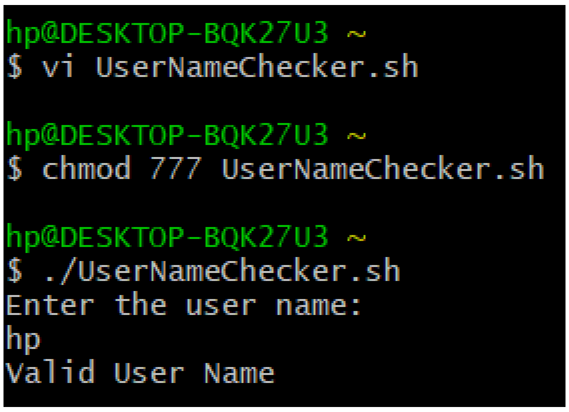
Q5) Write a script to display whether a user is valid or not.

Code:

```
#!/bin/bash
echo Enter the user name:
read uname
```

```
if [ "$uname" = "$USER" ]

then

echo Valid User Name

else

echo Invalid User Name

fi
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi UserNameChecker.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 UserNameChecker.sh

hp@DESKTOP-BQK27U3 ~
$ ./UserNameChecker.sh
Enter the user name:
hp
Valid User Name
```

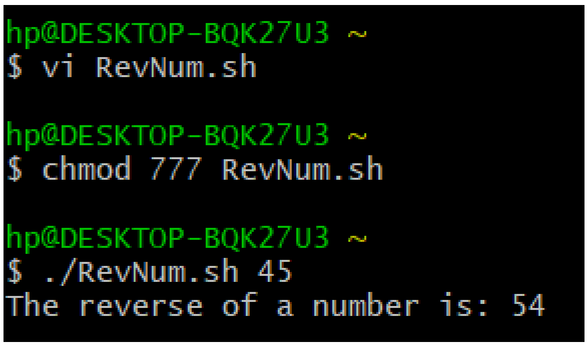
Experiment 7

Q1) Write a script to reverse number passed using positional parameter.

Code:

```
#!/bin/bash
a=`expr $1`
num=`expr $a`
d=0
until [ $num -eq 0 ]
do
r=`expr $num % 10`
b=`expr $d \* 10`
d=`expr $b + $r`
num=`expr $num / 10`
done
echo The reverse of a number is: $d
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi RevNum.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 RevNum.sh

hp@DESKTOP-BQK27U3 ~
$ ./RevNum.sh 45
The reverse of a number is: 54
```

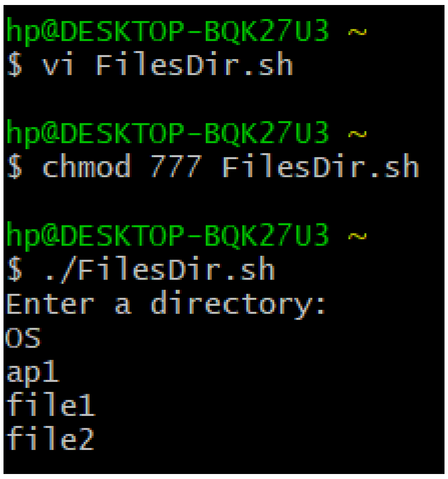
Q2) Write a script to list all files in a directory using for loop.

Code:

```
#!/bin/bash
echo Enter a directory:
read dire
cd $dire
for dire in `ls`
do
echo $dire
```

done

Output:

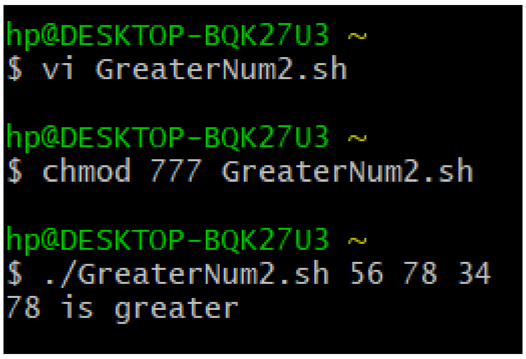

```
hp@DESKTOP-BQK27U3 ~
$ vi FilesDir.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 FilesDir.sh

hp@DESKTOP-BQK27U3 ~
$ ./FilesDir.sh
Enter a directory:
OS
ap1
file1
file2
```

Q3) Write a script to find largest value passed using command line.

Code:

```
#!/bin/bash

if [ $1 -gt $2 -a $1 -gt $3 ]

then

echo "$1 is greater"

elif [ $2 -gt $1 -a $2 -gt $3 ]

then

echo "$2 is greater"

else

echo "$3 is greater"

fi
```

Output:

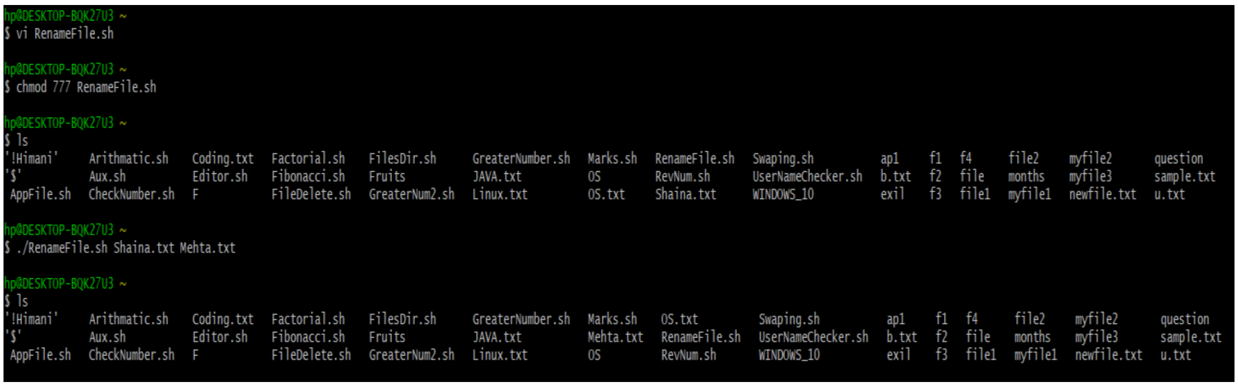

```
hp@DESKTOP-BQK27U3 ~
$ vi GreaterNum2.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 GreaterNum2.sh

hp@DESKTOP-BQK27U3 ~
$ ./GreaterNum2.sh 56 78 34
78 is greater
```

Q4) Write a script to search a particular file and rename it.

Code:

```
#!/bin/bash

dire=`expr $1`

a=`expr $2`

if [ "$(ls | grep $dire)" = "$dire" ]

then

mv $dire $a

else

echo File does not exist.

fi
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi RenameFile.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 RenameFile.sh

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'    Arithmatic.sh   Coding.txt  Factorial.sh   FilesDir.sh     GreaterNumber.sh  Marks.sh   RenameFile.sh  Swaping.sh           ap1    f1  f4     file2    myfile2      question
'$'          Aux.sh          Editor.sh   Fibonacci.sh   Fruits          JAVA.txt          OS         RevNum.sh      UserNameChecker.sh   b.txt  f2  file   months   myfile3      sample.txt
 AppFile.sh  CheckNumber.sh  F           FileDelete.sh  GreaterNum2.sh  Linux.txt         OS.txt     Shaina.txt     WINDOWS_10           exil   f3  file1  myfile1  newfile.txt  u.txt

hp@DESKTOP-BQK27U3 ~
$ ./RenameFile.sh Shaina.txt Mehta.txt

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'    Arithmatic.sh   Coding.txt  Factorial.sh   FilesDir.sh     GreaterNumber.sh  Marks.sh   OS.txt         Swaping.sh           ap1    f1  f4     file2    myfile2      question
'$'          Aux.sh          Editor.sh   Fibonacci.sh   Fruits          JAVA.txt          Mehta.txt  RenameFile.sh  UserNameChecker.sh   b.txt  f2  file   months   myfile3      sample.txt
 AppFile.sh  CheckNumber.sh  F           FileDelete.sh  GreaterNum2.sh  Linux.txt         OS         RevNum.sh      WINDOWS_10           exil   f3  file1  myfile1  newfile.txt  u.txt
```

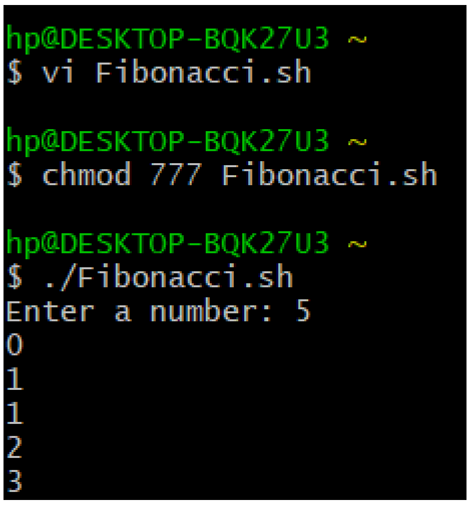
Q5) Write a script to print Fibonacci series.

Code:

```
#!/bin/bash

read -p "Enter a number: " a

b=0

c=1

echo $b

echo $c

d=`expr $a - 2`

for ((i=0; i<$d; i++))

do

e=`expr $b + $c`

echo $e
```

```
b=`expr $c`

c=`expr $e`

done
```

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi Fibonacci.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 Fibonacci.sh

hp@DESKTOP-BQK27U3 ~
$ ./Fibonacci.sh
Enter a number: 5
0
1
1
2
3
```

Q6) Write a script to create and delete directory using case.

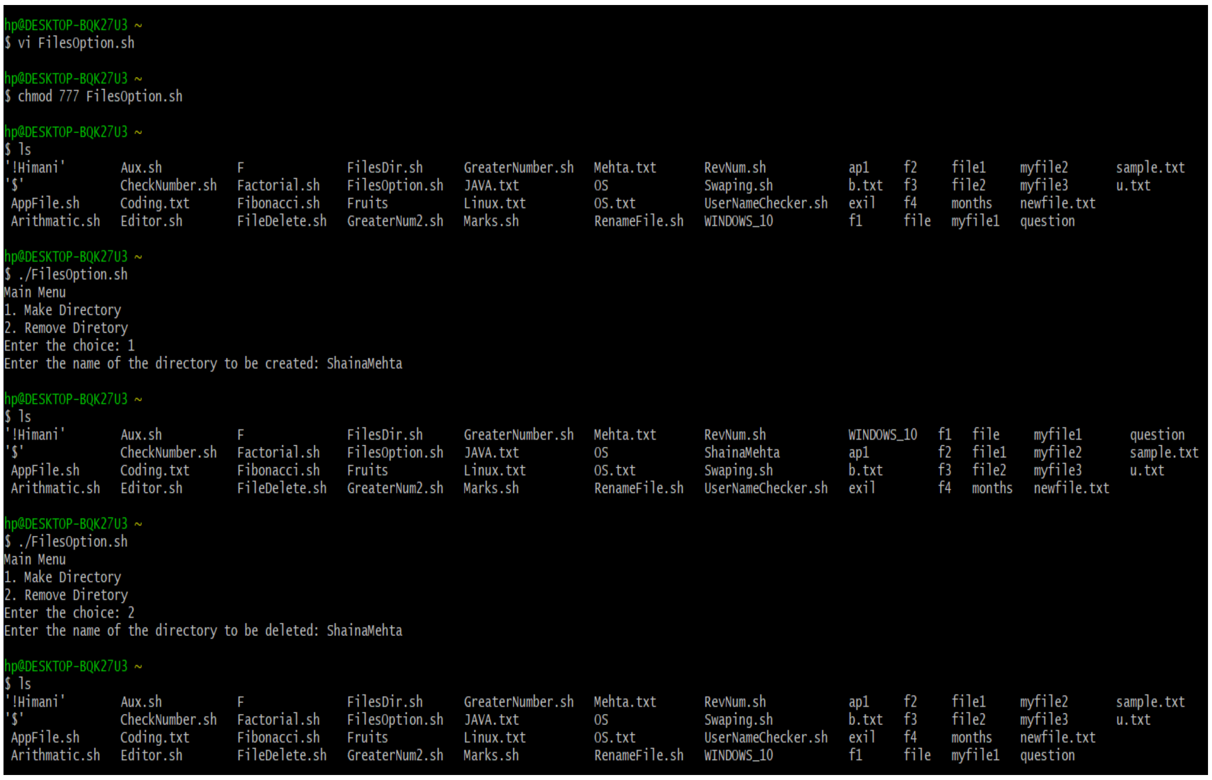
Code:

```
#!/bin/bash

echo Main Menu

echo 1. Make Directory

echo 2. Remove Diretory

read -p "Enter the choice: " ch

case $ch in

1)

read -p "Enter the name of the directory to be created: " mdire

mkdir $mdire

;;

2)

read -p "Enter the name of the directory to be deleted: " remdire

rmdir $remdire

;;

*)

echo Invalid Choice
```

;;

esac

Output:



```
hp@DESKTOP-BQK27U3 ~
$ vi FilesOption.sh

hp@DESKTOP-BQK27U3 ~
$ chmod 777 FilesOption.sh

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'        Aux.sh           F               FilesDir.sh      GreaterNumber.sh   Mehta.txt       RevNum.sh            ap1     f2    file1    myfile2       sample.txt
'$'              CheckNumber.sh   Factorial.sh    FilesOption.sh   JAVA.txt           OS              Swaping.sh           b.txt   f3    file2    myfile3       u.txt
 AppFile.sh      Coding.txt       Fibonacci.sh    Fruits           Linux.txt          OS.txt          UserNameChecker.sh   exil    f4    months   newfile.txt
 Arithmatic.sh   Editor.sh        FileDelete.sh   GreaterNum2.sh   Marks.sh           RenameFile.sh   WINDOWS_10           f1      file  myfile1  question

hp@DESKTOP-BQK27U3 ~
$ ./FilesOption.sh
Main Menu
1. Make Directory
2. Remove Diretory
Enter the choice: 1
Enter the name of the directory to be created: ShainaMehta

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'        Aux.sh           F               FilesDir.sh      GreaterNumber.sh   Mehta.txt       RevNum.sh            WINDOWS_10   f1   file    myfile1       question
'$'              CheckNumber.sh   Factorial.sh    FilesOption.sh   JAVA.txt           OS              ShainaMehta          ap1          f2   file1   myfile2       sample.txt
 AppFile.sh      Coding.txt       Fibonacci.sh    Fruits           Linux.txt          OS.txt          Swaping.sh           b.txt        f3   file2   myfile3       u.txt
 Arithmatic.sh   Editor.sh        FileDelete.sh   GreaterNum2.sh   Marks.sh           RenameFile.sh   UserNameChecker.sh   exil         f4   months  newfile.txt

hp@DESKTOP-BQK27U3 ~
$ ./FilesOption.sh
Main Menu
1. Make Directory
2. Remove Diretory
Enter the choice: 2
Enter the name of the directory to be deleted: ShainaMehta

hp@DESKTOP-BQK27U3 ~
$ ls
'!Himani'        Aux.sh           F               FilesDir.sh      GreaterNumber.sh   Mehta.txt       RevNum.sh            ap1     f2    file1    myfile2       sample.txt
'$'              CheckNumber.sh   Factorial.sh    FilesOption.sh   JAVA.txt           OS              Swaping.sh           b.txt   f3    file2    myfile3       u.txt
 AppFile.sh      Coding.txt       Fibonacci.sh    Fruits           Linux.txt          OS.txt          UserNameChecker.sh   exil    f4    months   newfile.txt
 Arithmatic.sh   Editor.sh        FileDelete.sh   GreaterNum2.sh   Marks.sh           RenameFile.sh   WINDOWS_10           f1      file  myfile1  question
```

Experiment 8

Date: 18-02-2021

Aim: To simulate FCFS scheduling algorithm using C programming language.

Software Used: Code Blocks IDE

Code:

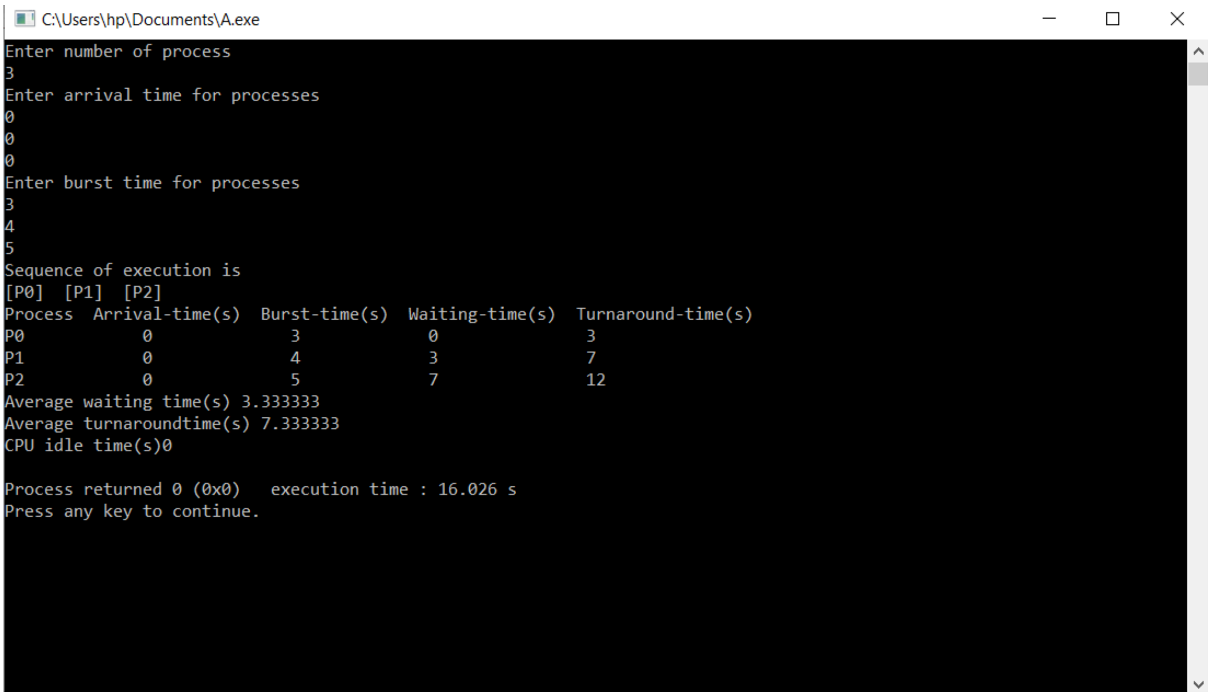
```
#include<stdio.h>
int main()
{
int at[10], at2[10], bt[100], ex[100], seq[100], re[100], wt[100];
int tat[100];
int n, i, j, start, position, max_time=0, min_time, idle=0, k=0;
float av1=0, av2=0;
printf("Enter number of process\n");
scanf("%d",&n);
printf("Enter arrival time for processes\n");
for(i=0;i<n;i++)
{
scanf("%d",&at[i]);
at2[i]=at[i];
}
printf("Enter burst time for processes\n");
for(i=0;i<n;i++)
{
scanf("%d",&bt[i]);
}
start=at[0];
for(i=1;i<n;i++)
{
if(start>at[i])
{
```

```
start=at[i];
}
}
printf("Sequence of execution is\n");
for(i=0;i<n;i++)
{
if(max_time<at[i])
{
max_time=at[i];
}
}
max_time=max_time+1;
for(i=0;i<n;i++,k++)
{  min_time=max_time;
for(j=0;j<n;j++){
if(at[j]!=-1)
{
if(at[j]<min_time)
{
min_time=at[j];
position=j;
}
} }
printf("[P%d]  ",position);
seq[k]=position;
if(start<at[position]){
re[position]=start;
idle+=at[position]-start;
start=at[position];
start+=bt[position];
at[position]=-1;
```

```
ex[position]=start;
}
else{
re[position]=start;
start+=bt[position];
at[position]=-1;
ex[position]=start;
}
}
printf("\n");
for(i=0;i<n;i++)
{
tat[i]=ex[i]-at2[i];
wt[i]=tat[i]-bt[i];
}
printf("Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)  Turnaround-time(s)\n");
for(i=0;i<n;i++)
{
printf("P%d           %d              %d             %d               %d\n", i, at2[i] ,bt[i] , wt[i], tat[i]);
}
for(i=0;i<n;i++)
{
av1+=tat[i];
av2+=wt[i];
}
printf("Average waiting time(s) %f\nAverage turnaroundtime(s) %f\nCPU idle time(s)%d\n",av2/n,av1/n,idle);
}
```

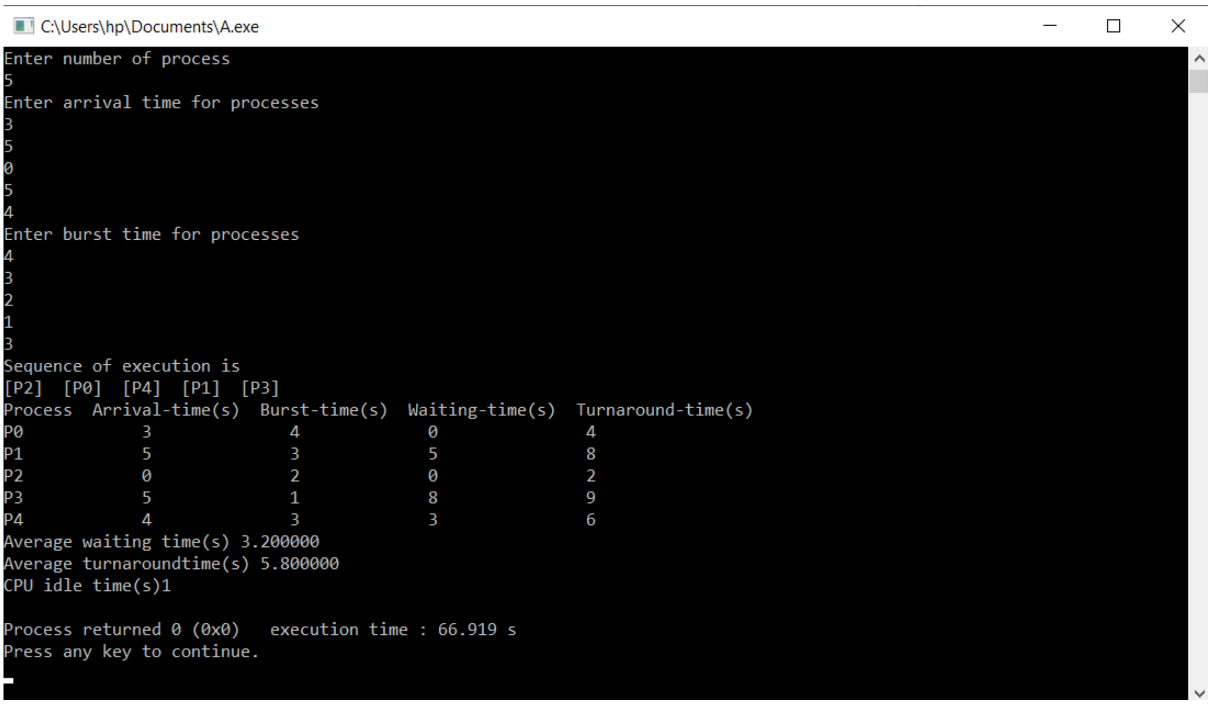
Output:

Case 1: Zero Arrival Time



```
Enter number of process
3
Enter arrival time for processes
0
0
0
Enter burst time for processes
3
4
5
Sequence of execution is
[P0]  [P1]  [P2]
Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)  Turnaround-time(s)
P0            0              3             0               3
P1            0              4             3               7
P2            0              5             7               12
Average waiting time(s) 3.333333
Average turnaroundtime(s) 7.333333
CPU idle time(s)0

Process returned 0 (0x0)   execution time : 16.026 s
Press any key to continue.
```

Case 2: Different Arrival Time



```
Enter number of process
5
Enter arrival time for processes
3
5
0
5
4
Enter burst time for processes
4
3
2
1
3
Sequence of execution is
[P2]  [P0]  [P4]  [P1]  [P3]
Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)  Turnaround-time(s)
P0            3              4             0               4
P1            5              3             5               8
P2            0              2             0               2
P3            5              1             8               9
P4            4              3             3               6
Average waiting time(s) 3.200000
Average turnaroundtime(s) 5.800000
CPU idle time(s)1

Process returned 0 (0x0)   execution time : 66.919 s
Press any key to continue.
```

Conclusion: The simulation of FCFS scheduling algorithm has been done successfully.

Experiment 9

Date: 05-03-2021

Aim: To simulate SJF scheduling algorithm using C programming language.

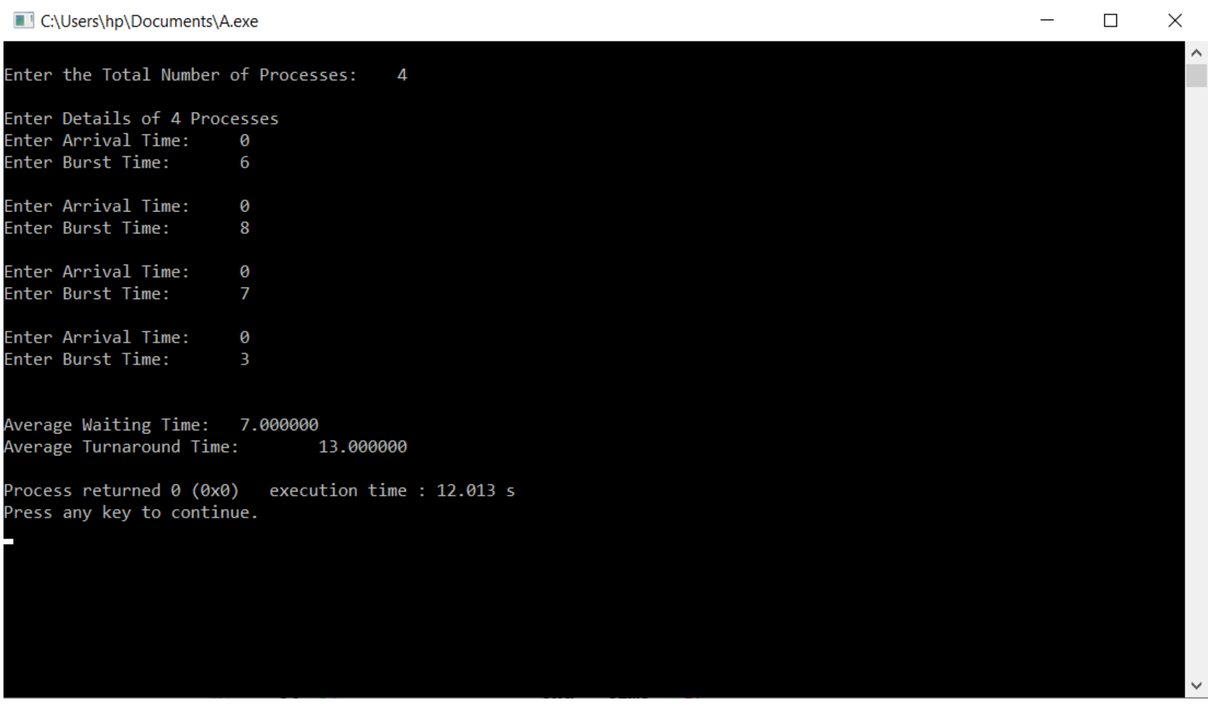
Software Used: Code Blocks IDE

Preemptive:

Code:

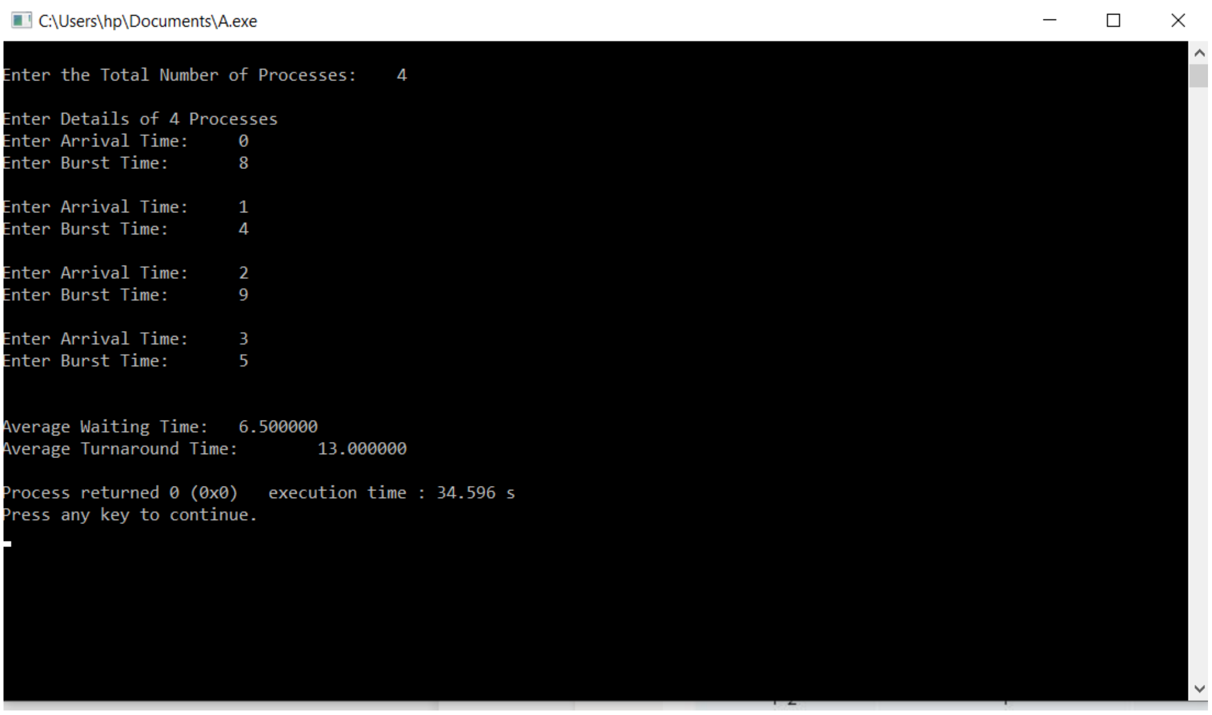
```
#include <stdio.h>
int main()
{
int at[10], bt[10], temp[10];
int i, smallest, count = 0, time, limit;
double wt = 0, tt = 0, end;
float avgWT, avgTT;
printf("\nEnter the Total Number of Processes:\t");
scanf("%d", &limit);
printf("\nEnter Details of %d Processes ", limit);
for(i = 0; i< limit; i++)
{
printf("\nEnter Arrival Time:\t");
scanf("%d", &at [i]);
printf("Enter Burst Time:\t");
scanf("%d", &bt [i]);
temp[i] = bt[i];
}
bt[9] = 9999;
for(time = 0; count != limit; time++)
{
smallest = 9;
for(i = 0; i< limit; i++)
{
if(at[i] <= time &&bt[i] <bt[smallest] &&bt[i] > 0)
{
smallest = i;
}
}
bt[smallest]--;
if(bt[smallest] == 0)
{
count++;
end = time + 1;
wt = wt + end - at[smallest] - temp[smallest];
tt = tt + end - at[smallest];
}
}
avgWT = wt / limit;
avgTT= tt / limit;
printf("\n\nAverage Waiting Time:\t%lf\n", avgWT);
printf("Average Turnaround Time:\t%lf\n", avgTT);
return 0;
}
```

<u>Output:</u>



C:\Users\hp\Documents\A.exe

```
Enter the Total Number of Processes:    4

Enter Details of 4 Processes
Enter Arrival Time:     0
Enter Burst Time:       6

Enter Arrival Time:     0
Enter Burst Time:       8

Enter Arrival Time:     0
Enter Burst Time:       7

Enter Arrival Time:     0
Enter Burst Time:       3


Average Waiting Time:   7.000000
Average Turnaround Time:        13.000000

Process returned 0 (0x0)   execution time : 12.013 s
Press any key to continue.
```



C:\Users\hp\Documents\A.exe

```
Enter the Total Number of Processes:    4

Enter Details of 4 Processes
Enter Arrival Time:     0
Enter Burst Time:       8

Enter Arrival Time:     1
Enter Burst Time:       4

Enter Arrival Time:     2
Enter Burst Time:       9

Enter Arrival Time:     3
Enter Burst Time:       5


Average Waiting Time:   6.500000
Average Turnaround Time:        13.000000

Process returned 0 (0x0)   execution time : 34.596 s
Press any key to continue.
```

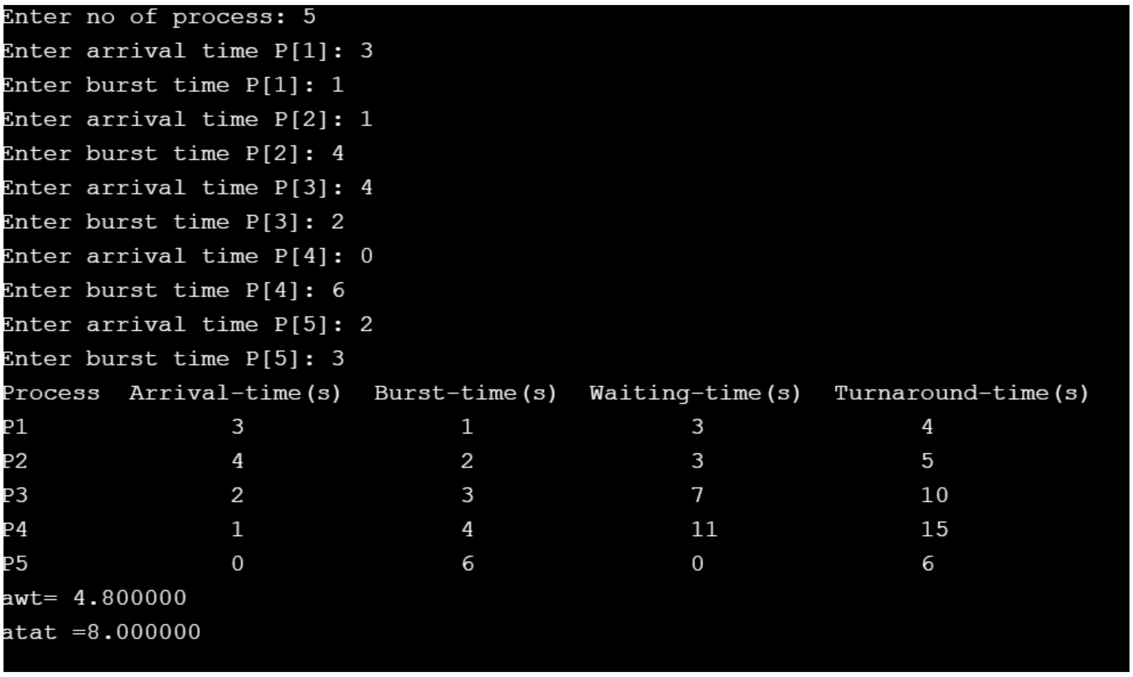
<u>Non - Preemptive:</u>

Code:

```
#include<stdio.h>

#include<conio.h>

void main()

{

int n,temp,tt=0,min,d,i,j;

float atat=0,awt=0,stat=0,swt=0;

printf("Enter no of process: ");

scanf("%d",&n);

int a[10],b[10],e[10],tat[10],wt[10];

for(i=0;i<n;i++)

{

printf("Enter arrival time P[%d]: ",i+1);

scanf("%d",&a[i]);

printf("Enter burst time P[%d]: ",i+1);

scanf("%d",&b[i]);

}

for(i=0;i<n;i++)

{

for(j=i+1;j<n;j++)

{

if(b[i]>b[j])

{

temp=a[i];

a[i]=a[j];

a[j]=temp;

temp=b[i];

b[i]=b[j];

b[j]=temp;

}

}
}
min=a[0];
for(i=0;i<n;i++)
{
if(min>a[i])
{
min=a[i];
d=i;
}
}
tt=min;
e[d]=tt+b[d];
tt=e[d];
for(i=0;i<n;i++)
{
if(a[i]!=min)
{
e[i]=b[i]+tt;
tt=e[i];
}
}
for(i=0;i<n;i++)
{
tat[i]=e[i]-a[i];
stat=stat+tat[i];
wt[i]=tat[i]-b[i];
swt=swt+wt[i];
}
atat=stat/n;
awt=swt/n;
printf("Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)  Turnaround-time(s)\n");
```

```
for(i=0;i<n;i++)
{
printf("P%d\t\t%d\t\t%d\t\t%d\t\t%d\n",i+1,a[i],b[i],wt[i],tat[i]);
}
printf("awt= %f\natat =%f",awt,atat);
getch();
}
```

Output:



```
Enter no of process: 5
Enter arrival time P[1]: 3
Enter burst time P[1]: 1
Enter arrival time P[2]: 1
Enter burst time P[2]: 4
Enter arrival time P[3]: 4
Enter burst time P[3]: 2
Enter arrival time P[4]: 0
Enter burst time P[4]: 6
Enter arrival time P[5]: 2
Enter burst time P[5]: 3
Process  Arrival-time(s)  Burst-time(s)  Waiting-time(s)  Turnaround-time(s)
P1              3               1               3               4
P2              4               2               3               5
P3              2               3               7               10
P4              1               4               11              15
P5              0               6               0               6
awt= 4.800000
atat =8.000000
```

Conclusion: The simulation of SJF scheduling algorithm has been done successfully.

Experiment 10

Date: 24-03-2021

Aim: To simulate Round Robin scheduling algorithm using C programming language.

Software Used: Code Blocks IDE

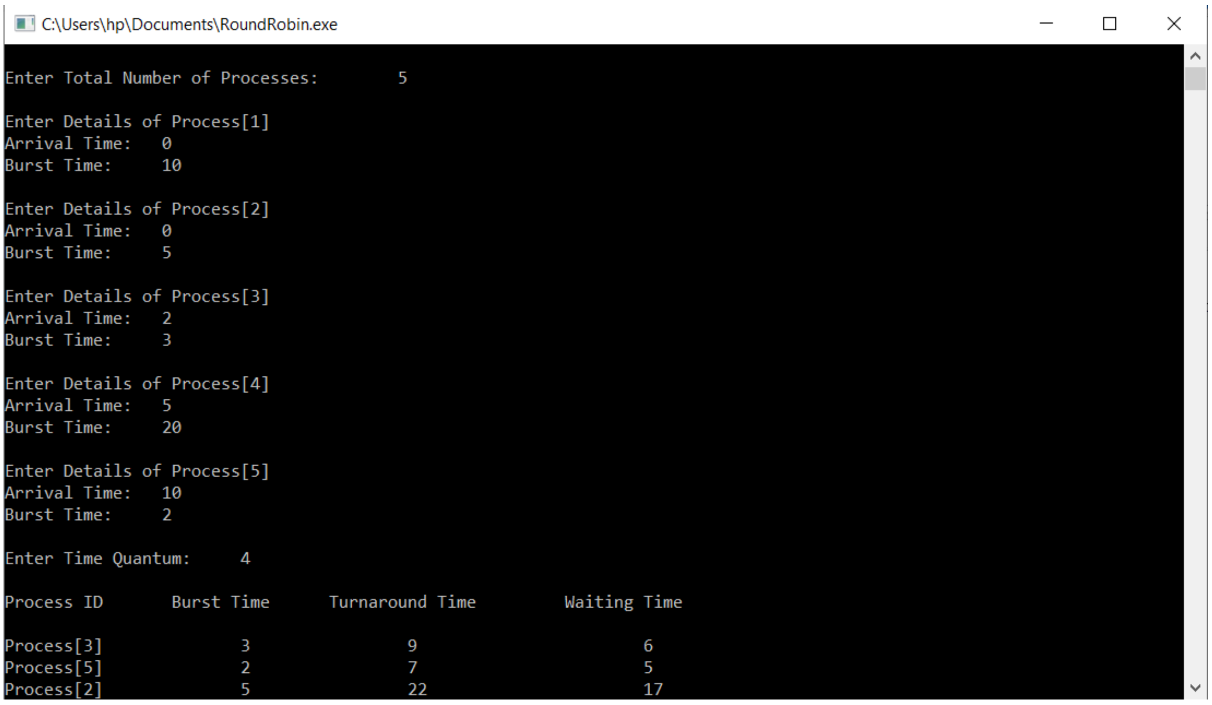
Code:

```
#include<stdio.h>
int main()
{
int i, limit, total = 0, x, counter = 0, tq;
int wt = 0, tt = 0, at[10], bt[10], temp[10];
float avgWT, avgTT;
printf("\nEnter Total Number of Processes:\t");
scanf("%d", &limit);
x = limit;
for(i = 0; i < limit; i++)
{
printf("\nEnter Details of Process[%d]\n", i + 1);
printf("Arrival Time:\t");
scanf("%d", &at[i]);
printf("Burst Time:\t");
scanf("%d", &bt[i]);
temp[i] = bt[i];
}
printf("\nEnter Time Quantum:\t");
scanf("%d", &tq);
printf("\nProcess ID \t Burst Time \t Turnaround Time \t Waiting Time\n");
for(total = 0, i = 0; x != 0;)
{
if(temp[i] <= tq && temp[i] > 0)
{
```

```
total = total + temp[i];
temp[i] = 0;
counter = 1;
}
else if(temp[i] > 0)
{
temp[i] = temp[i] - tq;
total = total + tq;
}
if(temp[i] == 0 && counter == 1)
{
x--;
printf("\nProcess[%d]\t\t%d\t\t %d\t\t\t %d", i + 1, bt[i], total - at[i], total - at[i] - bt[i]);
wt = wt + total - at[i] - bm[i];
tt = tt + total - at[i];
counter = 0;
}
if(i == limit - 1)
{
i = 0;
}
else if(at[i + 1] <= total)
{
i++;
}
else
{
i = 0;
}
}
avgWT = wt * 1.0 / limit;
```

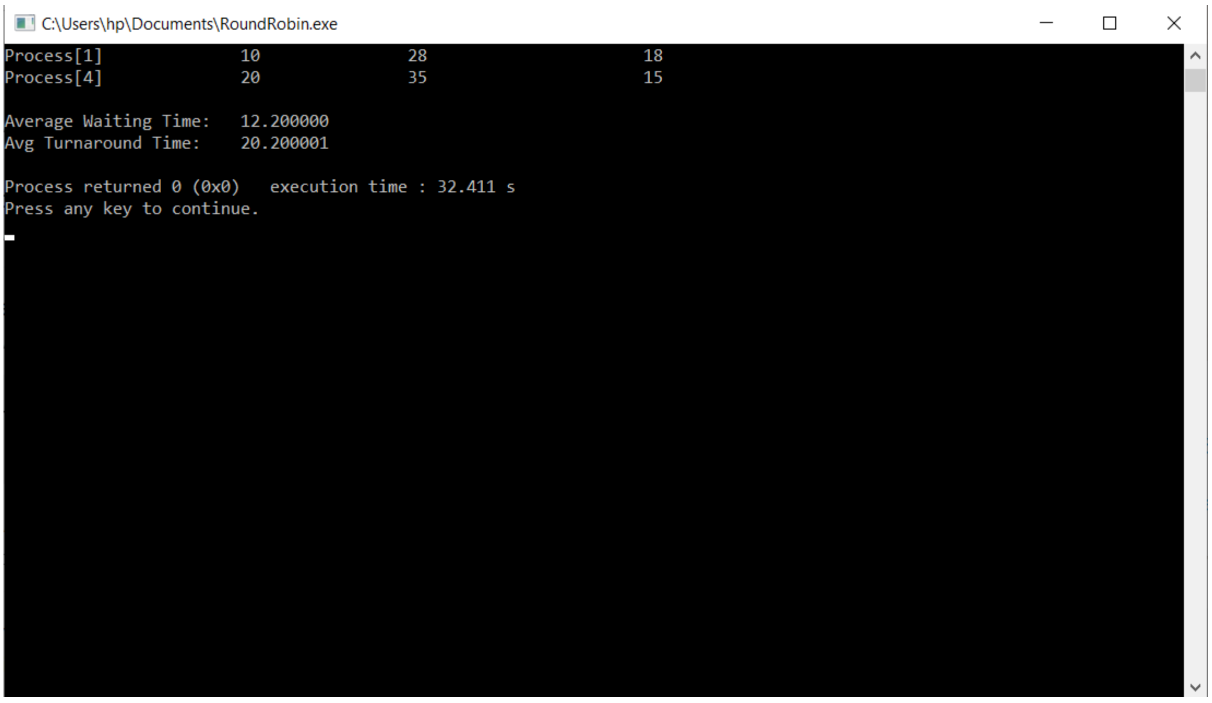
```
avgTT = tt * 1.0 / limit;
printf("\n\nAverage Waiting Time:\t%f", avgWT);
printf("\nAvg Turnaround Time:\t%f\n", avgTT);
return 0;
}
```

Output:



C:\Users\hp\Documents\RoundRobin.exe

```
Enter Total Number of Processes:        5

Enter Details of Process[1]
Arrival Time:   0
Burst Time:     10

Enter Details of Process[2]
Arrival Time:   0
Burst Time:     5

Enter Details of Process[3]
Arrival Time:   2
Burst Time:     3

Enter Details of Process[4]
Arrival Time:   5
Burst Time:     20

Enter Details of Process[5]
Arrival Time:   10
Burst Time:     2

Enter Time Quantum:     4

Process ID       Burst Time      Turnaround Time         Waiting Time

Process[3]              3               9                       6
Process[5]              2               7                       5
Process[2]              5               22                      17
```



C:\Users\hp\Documents\RoundRobin.exe

```
Process[1]              10              28                      18
Process[4]              20              35                      15

Average Waiting Time:   12.200000
Avg Turnaround Time:    20.200001

Process returned 0 (0x0)   execution time : 32.411 s
Press any key to continue.
```

Conclusion: The simulation of Round Robin scheduling algorithm has been done successfully.

Experiment 11

Date: 24-03-2021

Aim: To simulate Banker's algorithm using C programming language.

Software Used: Code Blocks IDE

Code:

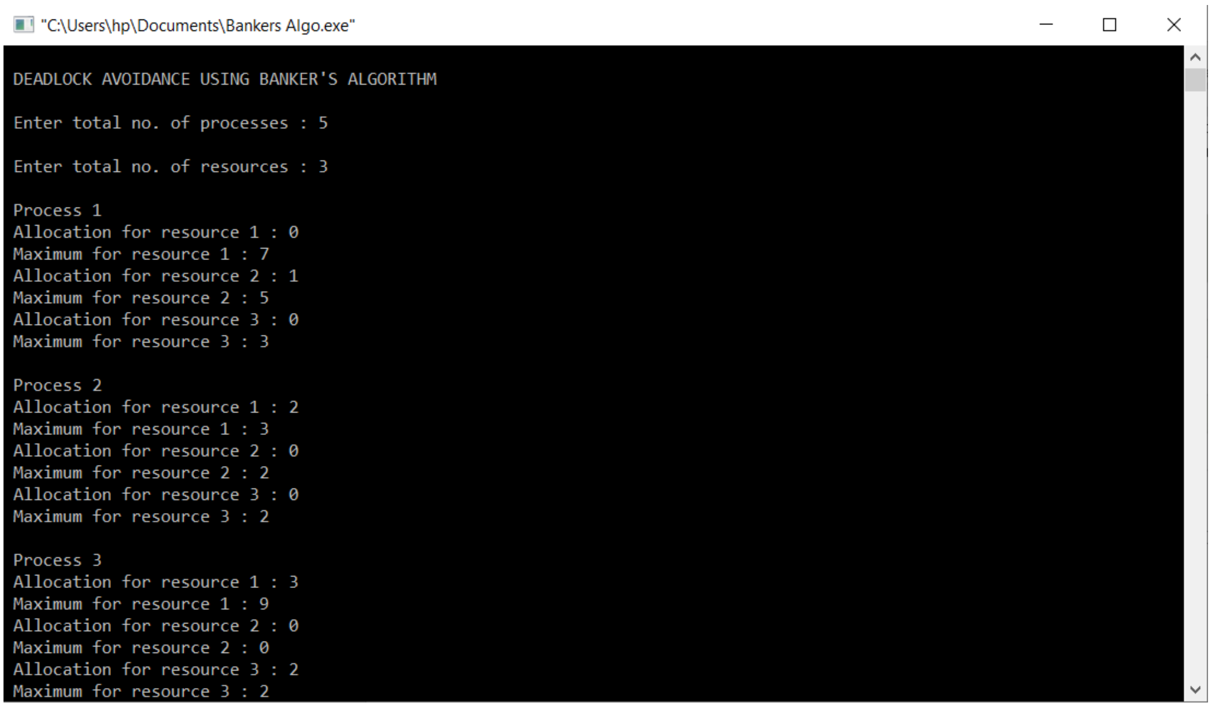
```
#include<stdio.h>
#include<stdlib.h>
void print(int x[][10],int n,int m){
for(int i=0;i<n;i++){
printf("\n");
for(int j=0;j<m;j++){
printf("%d\t",x[i][j]);
}
}
}
void resRequest(int A[10][10],int N[10][10],int AV[10][10],int pid,int m)
{
int reqmat[1][10];
printf("\n Enter additional request :- \n");
for(int i=0;i<m;i++){
printf(" Request for resource %d : ",i+1);
scanf("%d",&reqmat[0][i]);
}
for(int i=0;i<m;i++)
if(reqmat[0][i] > N[pid][i]){
printf("\n Error encountered.\n");
exit(0);
}
for(int i=0;i<m;i++)
if(reqmat[0][i] > AV[0][i]){
```

```
printf("\n Resources unavailable.\n");
exit(0);
}
for(int i=0;i<m;i++){
AV[0][i]-=reqmat[0][i];
A[pid][i]+=reqmat[0][i];
N[pid][i]-=reqmat[0][i];
}
}
int safetyCheck(int A[][10],int N[][10],int AV[1][10],int n,int m,int a[]){
int x=0;
int F[10],W[1][10];
int pf=0,f=0;
for(int i=0;i<n;i++)
F[i]=0;
for(int i=0;i<m;i++)
W[0][i]=AV[0][i];
for(int k=0;k<n;k++){
for(int i=0;i<n;i++){
if(F[i] == 0){
f=0;
for(int j=0;j<m;j++){
if(N[i][j] > W[0][j])
f=1;
}
if(f == 0 && F[i] == 0){
for(int j=0;j<m;j++)
W[0][j]+=A[i][j];
F[i]=1;
pf++;
a[x++]=i;
}
}
}
if(pf == n)
return 1;
}
return 0;
}
void accept(int A[][10],int N[][10],int M[10][10],int W[1][10],int *n,int *m){
printf("\n Enter total no. of processes : ");
scanf("%d",n);
printf("\n Enter total no. of resources : ");
scanf("%d",m);
for(int i=0;i<*n;i++){
printf("\n Process %d\n",i+1);
for(int j=0;j<*m;j++){
printf(" Allocation for resource %d : ",j+1);
scanf("%d",&A[i][j]);
printf(" Maximum for resource %d : ",j+1);
scanf("%d",&M[i][j]);
}
}
printf("\n Available resources : \n");
for(int i=0;i<*m;i++){
printf(" Resource %d : ",i+1);
scanf("%d",&W[0][i]);
}
for(int i=0;i<*n;i++)
for(int j=0;j<*m;j++)
N[i][j]=M[i][j]-A[i][j];
printf("\n Allocation Matrix");
```

```
print(A,*n,*m);
printf("\n Maximum Requirement Matrix");
print(M,*n,*m);
printf("\n Need Matrix");
print(N,*n,*m);
}
int banker(int A[][10],int N[][10],int W[1][10],int n,int m){
int j,a[10];
j=safetyCheck(A,N,W,n,m,a);
if(j != 0 ){
printf("\n\n");
for(int i=0;i<n;i++)
printf(" P%d  ",a[i]);
printf("\n A safety sequence has been detected.\n");
return 1;
}else{
printf("\n Deadlock has occured.\n");
return 0;
}
}
int main(){
int ret;
int A[10][10];
int M[10][10];
int N[10][10];
int W[1][10];
int n,m,pid,ch;
printf("\n DEADLOCK AVOIDANCE USING BANKER'S ALGORITHM\n");
accept(A,N,M,W,&n,&m);
ret=banker(A,N,W,n,m);
if(ret !=0 ){
```

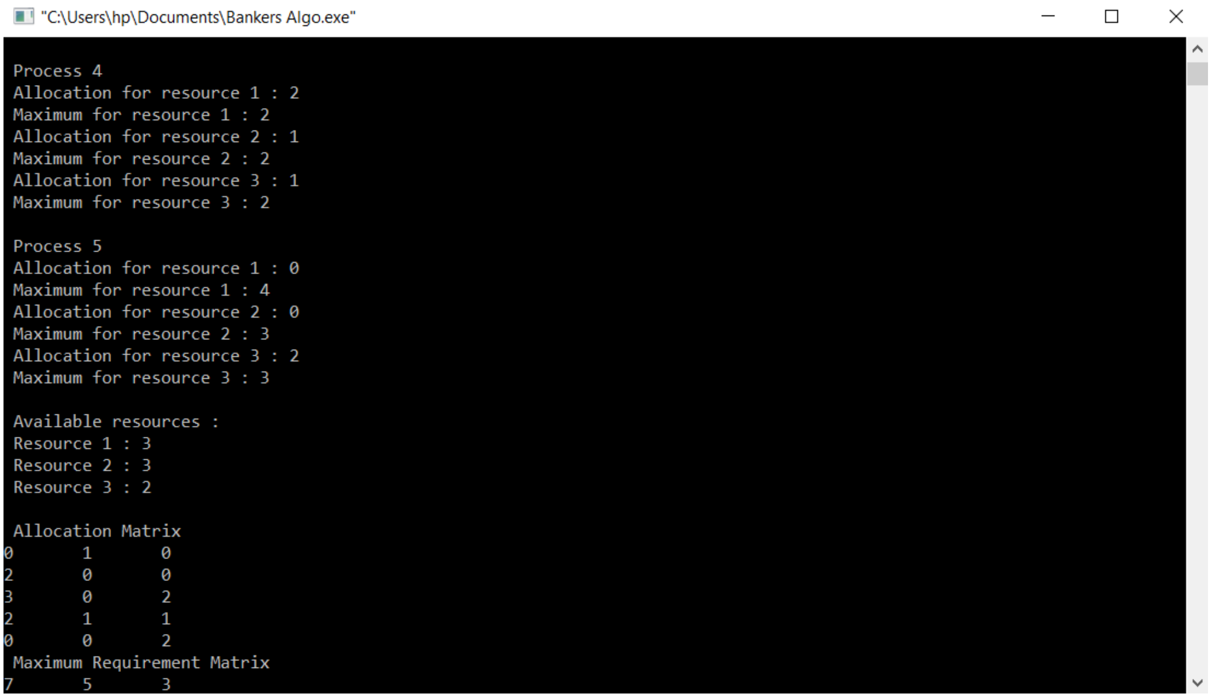
```
printf("\n Do you want make an additional request ? (1=Yes|0=No)");
scanf("%d",&ch);
if(ch == 1){
printf("\n Enter process no. : ");
scanf("%d",&pid);
resRequest(A,N,W,pid-1,m);
ret=banker(A,N,W,n,m);
if(ret == 0 )
exit(0);
}
}else
exit(0);
return 0;
}
```

<u>Output:</u>



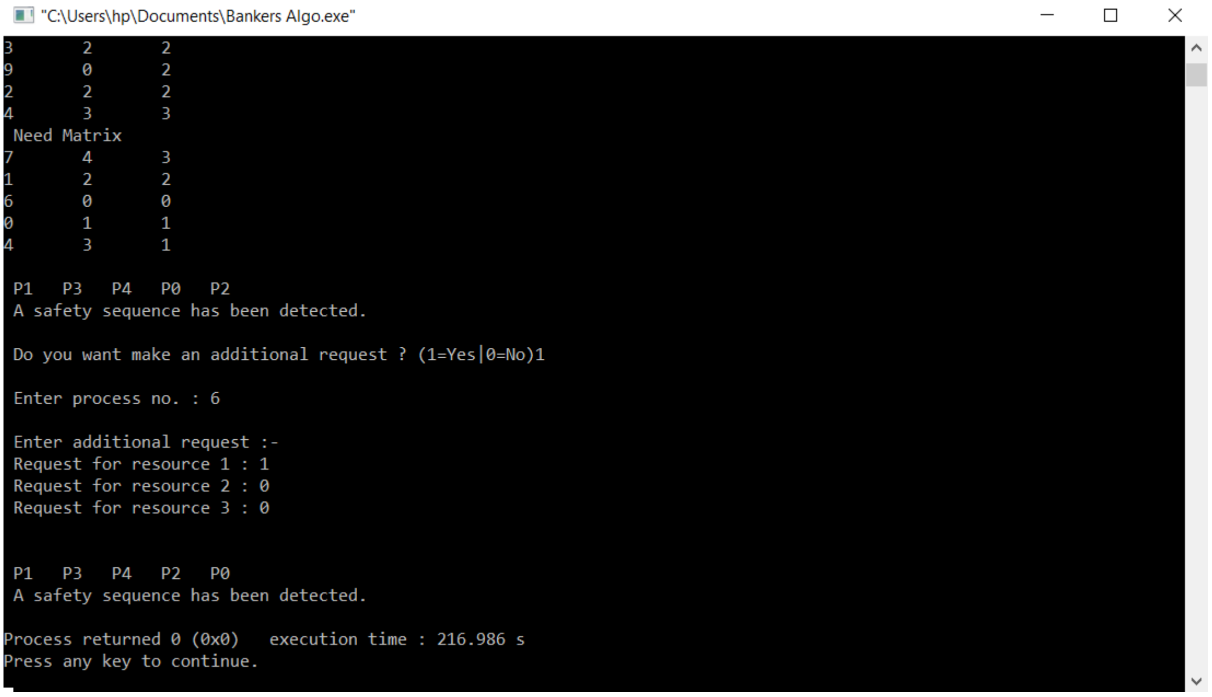
"C:\Users\hp\Documents\Bankers Algo.exe"

```
DEADLOCK AVOIDANCE USING BANKER'S ALGORITHM

Enter total no. of processes : 5

Enter total no. of resources : 3

Process 1
Allocation for resource 1 : 0
Maximum for resource 1 : 7
Allocation for resource 2 : 1
Maximum for resource 2 : 5
Allocation for resource 3 : 0
Maximum for resource 3 : 3

Process 2
Allocation for resource 1 : 2
Maximum for resource 1 : 3
Allocation for resource 2 : 0
Maximum for resource 2 : 2
Allocation for resource 3 : 0
Maximum for resource 3 : 2

Process 3
Allocation for resource 1 : 3
Maximum for resource 1 : 9
Allocation for resource 2 : 0
Maximum for resource 2 : 0
Allocation for resource 3 : 2
Maximum for resource 3 : 2
```



"C:\Users\hp\Documents\Bankers Algo.exe"

```
Process 4
Allocation for resource 1 : 2
Maximum for resource 1 : 2
Allocation for resource 2 : 1
Maximum for resource 2 : 2
Allocation for resource 3 : 1
Maximum for resource 3 : 2

Process 5
Allocation for resource 1 : 0
Maximum for resource 1 : 4
Allocation for resource 2 : 0
Maximum for resource 2 : 3
Allocation for resource 3 : 2
Maximum for resource 3 : 3

Available resources :
Resource 1 : 3
Resource 2 : 3
Resource 3 : 2

Allocation Matrix
0       1       0
2       0       0
3       0       2
2       1       1
0       0       2
Maximum Requirement Matrix
7       5       3
```



"C:\Users\hp\Documents\Bankers Algo.exe"

```
3       2       2
9       0       2
2       2       2
4       3       3
Need Matrix
7       4       3
1       2       2
6       0       0
0       1       1
4       3       1

P1   P3   P4   P0   P2
A safety sequence has been detected.

Do you want make an additional request ? (1=Yes|0=No)1

Enter process no. : 6

Enter additional request :-
Request for resource 1 : 1
Request for resource 2 : 0
Request for resource 3 : 0


P1   P3   P4   P2   P0
A safety sequence has been detected.

Process returned 0 (0x0)   execution time : 216.986 s
Press any key to continue.
```

Conclusion: The simulation of Banker’s algorithm has been done successfully.

Experiment 12

Date: 24-03-2021

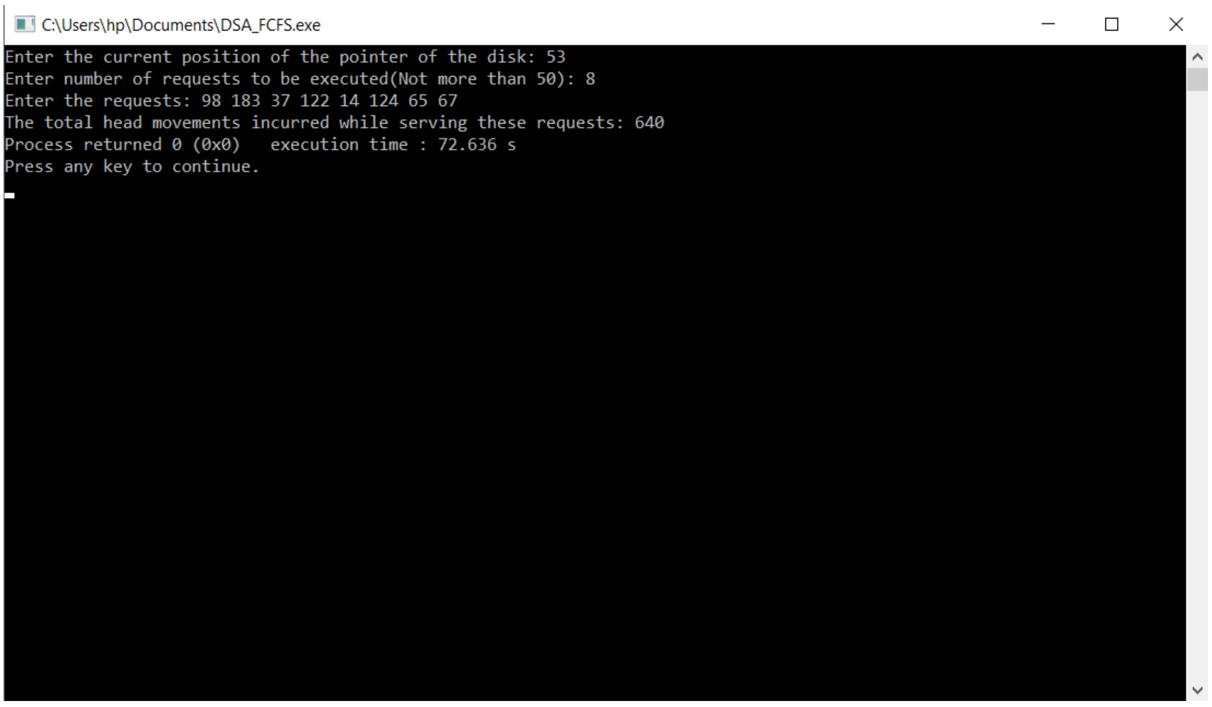
Aim: To simulate FIFO disk scheduling algorithm using C programming language.

Software Used: Code Blocks IDE

Code:

```
#include <stdio.h>
#include <math.h>
int main(){
int requests[51],rComp[50],n,total=0;
printf("Enter the current position of the pointer of the disk: ");
scanf("%d",&requests[0]);
printf("Enter number of requests to be executed(Not more than 50): ");
scanf("%d",&n);
printf("Enter the requests: ");
for(int i=1;i<=n;i++){
scanf("%d",&requests[i]);
}
for(int i=1,j=i-1;i<=n;i++,j++){
rComp[j]=abs(requests[j]-requests[i]);
total+=rComp[j];
}
printf("The total head movements incurred while serving these requests: %d",total);
return 0;
}
```

Output:



C:\Users\hp\Documents\DSA_FCFS.exe

```
Enter the current position of the pointer of the disk: 53
Enter number of requests to be executed(Not more than 50): 8
Enter the requests: 98 183 37 122 14 124 65 67
The total head movements incurred while serving these requests: 640
Process returned 0 (0x0)   execution time : 72.636 s
Press any key to continue.
```

Conclusion: The simulation of FIFO disk scheduling algorithm has been done successfully.