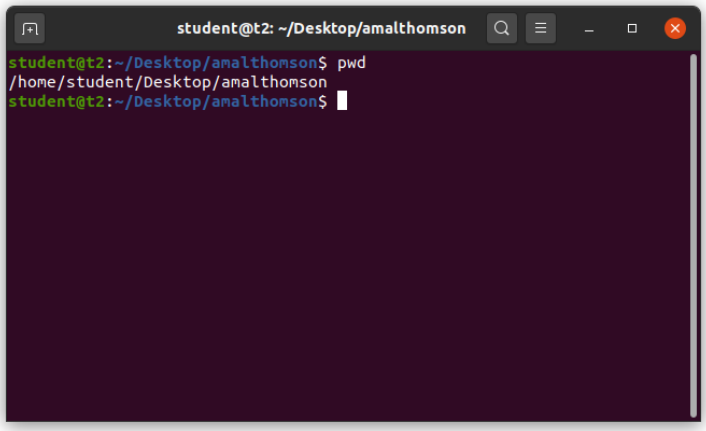
Experiment No.: 3

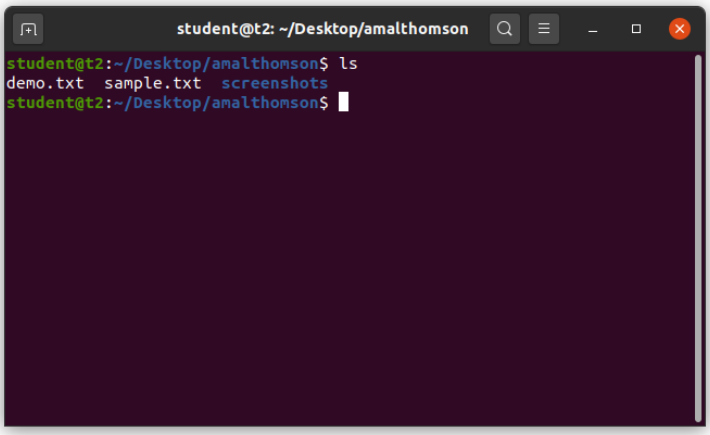
Aim
Familiarization of Linux Commands.

CO2
Perform system administration tasks.

Procedure
1. **pwd** – used to print the working directory. After execution it shows the absolute path.
   Syntax: $ pwd
   Output:

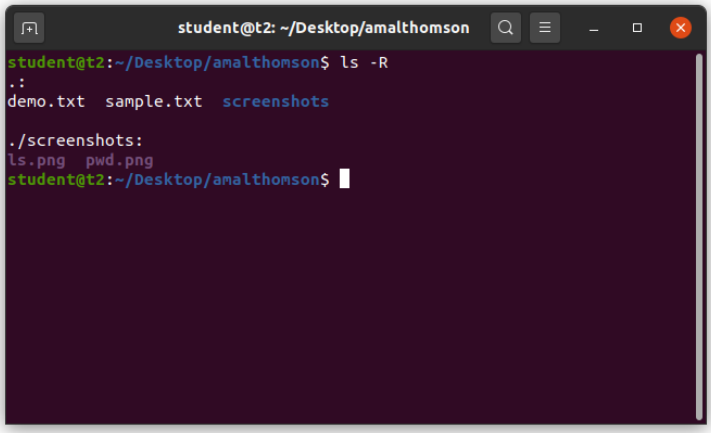


2. **ls** – used to list the files and content in the directory.
   Syntax: $ ls
   Output:

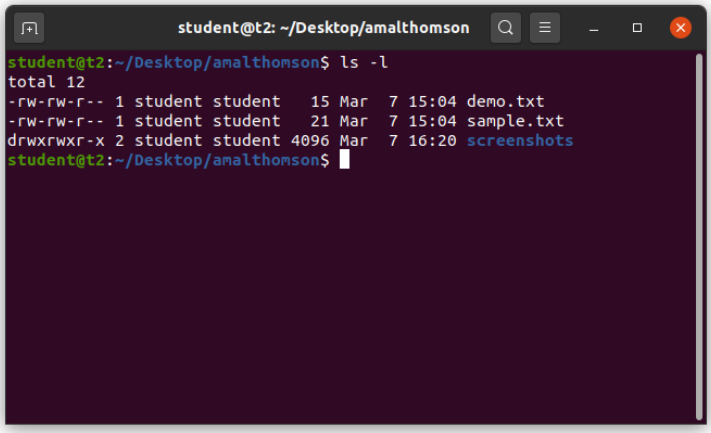


Options of ls command.

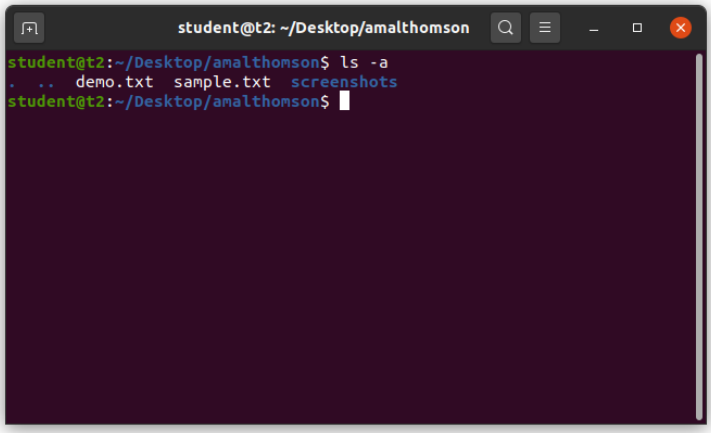
a) **ls -R** – used to list the directory as well as the subdirectory.
   Syntax: $ ls -R
   Output:



b) **ls -l** – used to view the long list of directory.
Syntax: $ ls -l
Output:

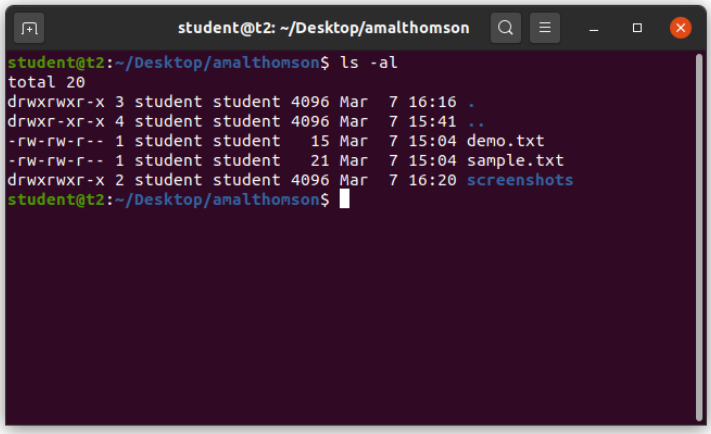


c) **ls -a** – used to view the list in directory along with hidden files.
Syntax: $ ls -a
Output:

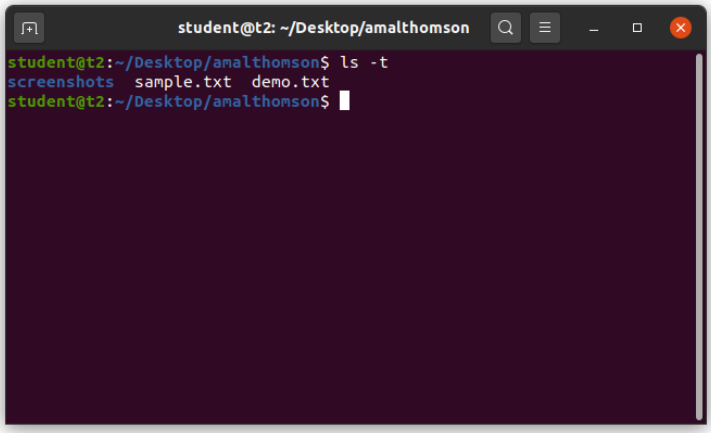


d) **ls -al** – used to view the list in directory with detailed information along with hidden files.
Syntax: $ ls -al

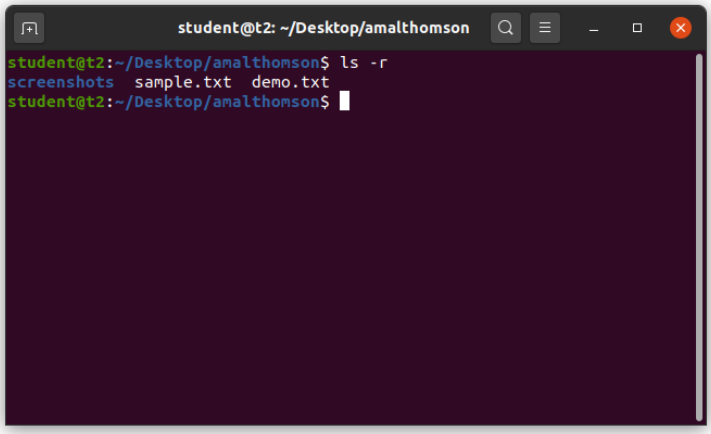
Output:



e) **ls -t** – used to view the list in sorted order of last modified.
Syntax: $ ls -t
Output:



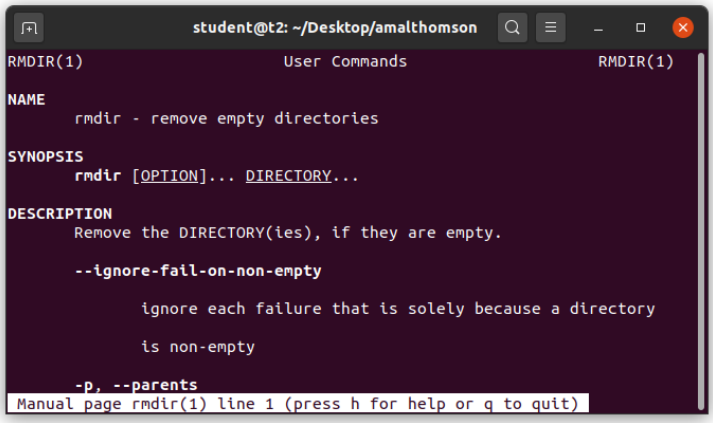
f) **ls -r** – used to view the list in reverse order of last modified.
Syntax: $ ls -r
Output:



3. **man** - used to learn and understand the existing commands we can learn and understand about different commands from the shell using man command.

Syntax: $ man mkdir

Output:



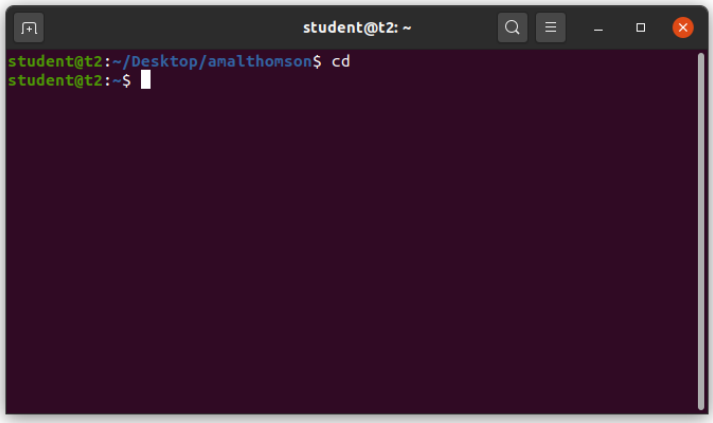
4. **cd** – used to navigate through directory.

   Options of cd commands:

   a) cd – used to switch to home directory.

      Syntax: $ cd

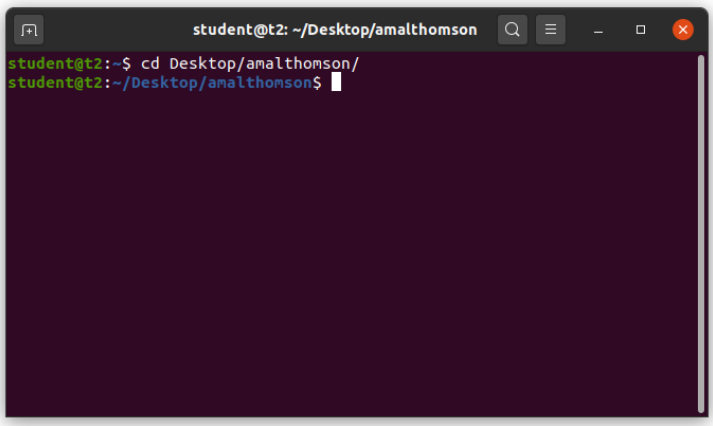
      Output:



   b) **cd <path>** - used to change to a particular path or directory

      Syntax: $ cd <*directory_path*>

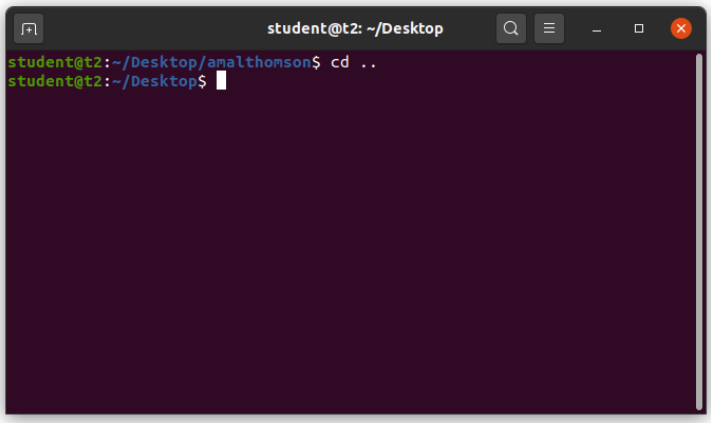
      Output:



c) **cd ..** – used to switch back to previous directory or one directory back from the current directory

Syntax: $ cd ..

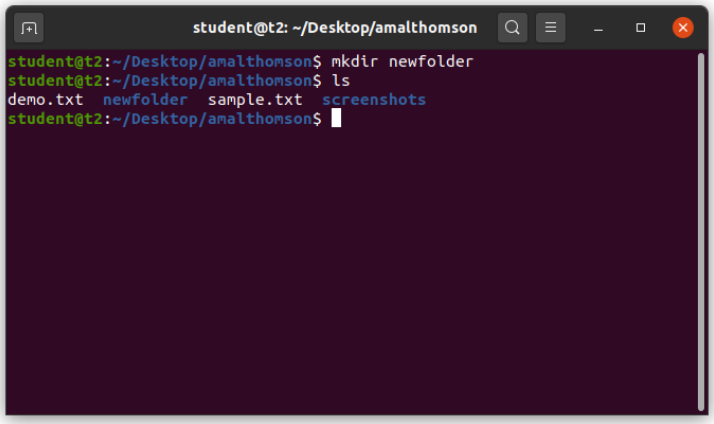
Output:



5. **mkdir** – Used to make new directory.

Syntax: $ mkdir <*directory_name*>

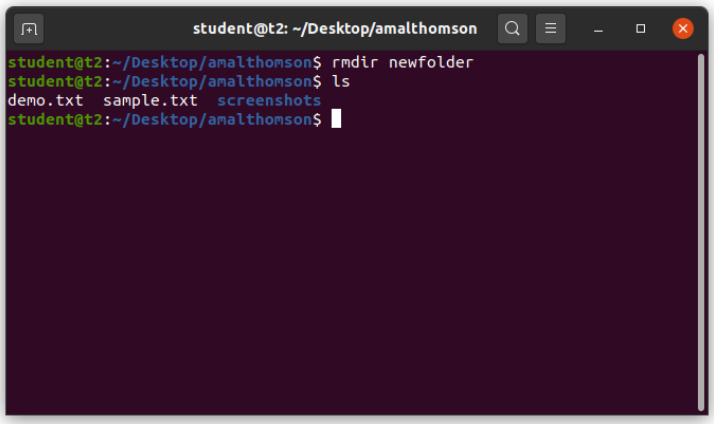
Output:



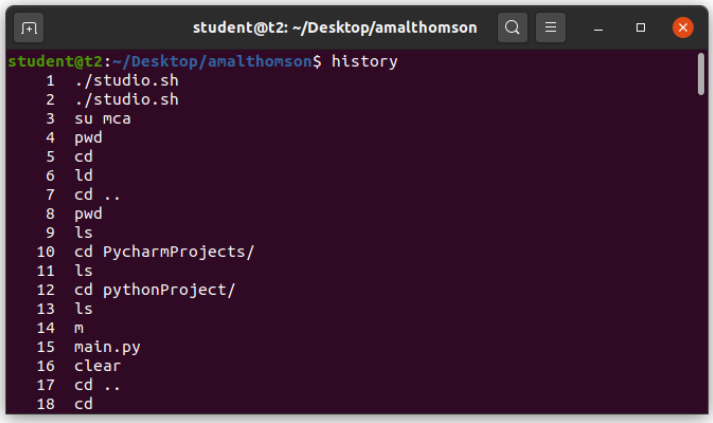
6. **rmdir** – used to remove a directory.

Syntax: $ rmdir <*directory_name*>

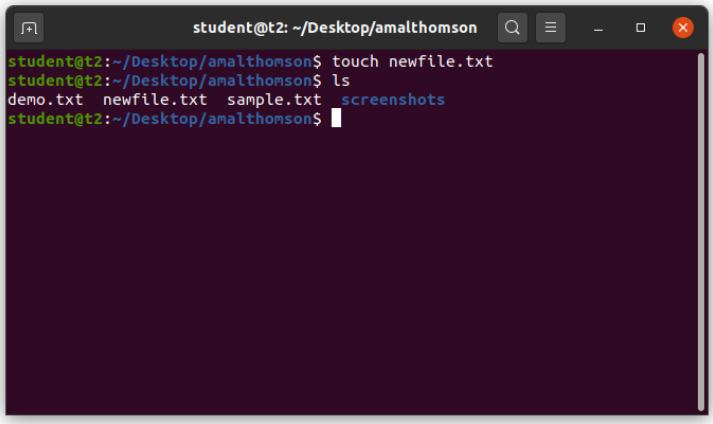
Output:



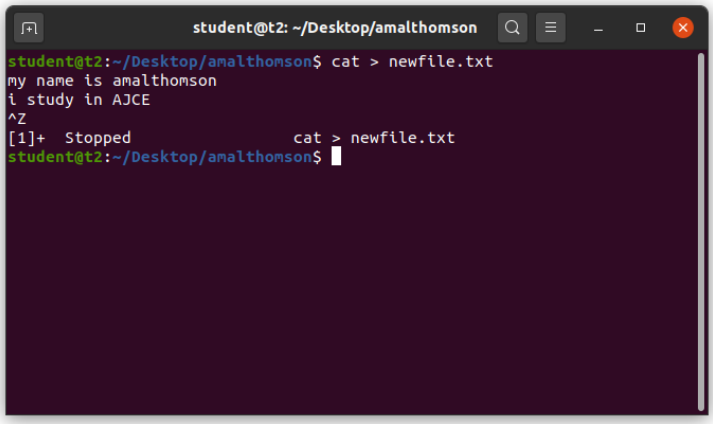
7. **history** – used to view the list of commands executed in a certain period of time.
   Syntax: $ history
   Output;



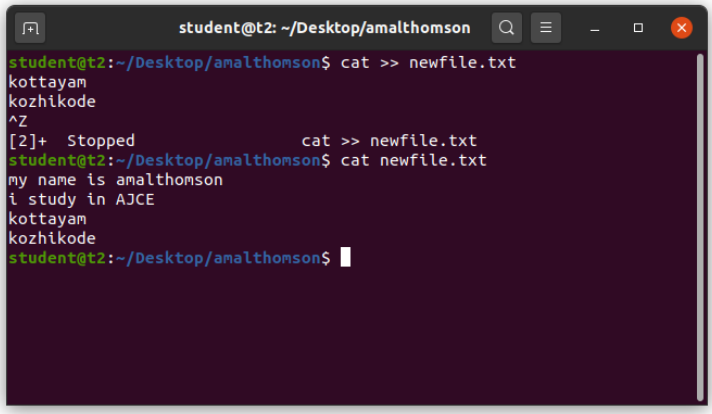
8. **touch** – used to create a new blank file.
   Syntax: $ touch *<filename>*
   Output:



9. **cat –** used to create a new blank file and also to add contents to the file.
   Options of cat commands:
   a) **cat >** – used to create a new blank file and also to add contents to the file.
      Syntax: $ cat > *<filename>*
      Output:

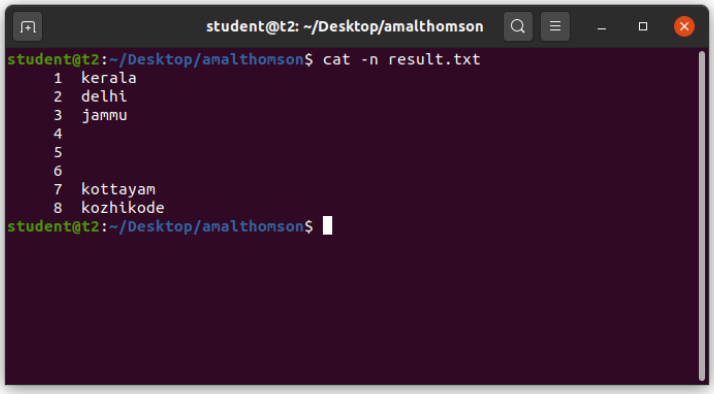


b) **cat >>** – used to append new contents to existing file.
Syntax: $ cat >> *<filename>*
Output:

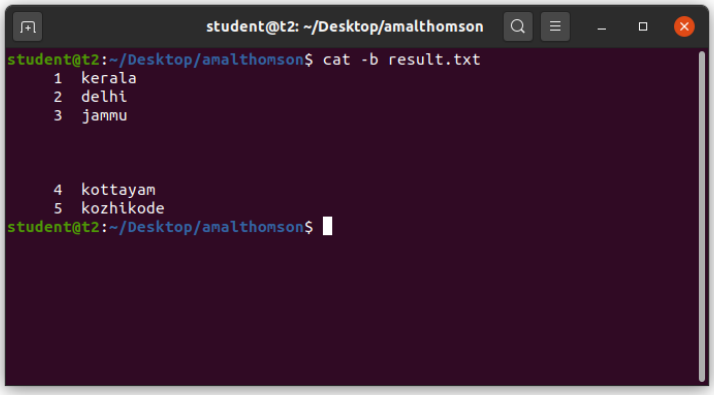


c) **cat file1 file2 > file3** – copy contents of two files to a third file.
Syntax: $ cat *<filename><filename>* > *<filename>*
Output:

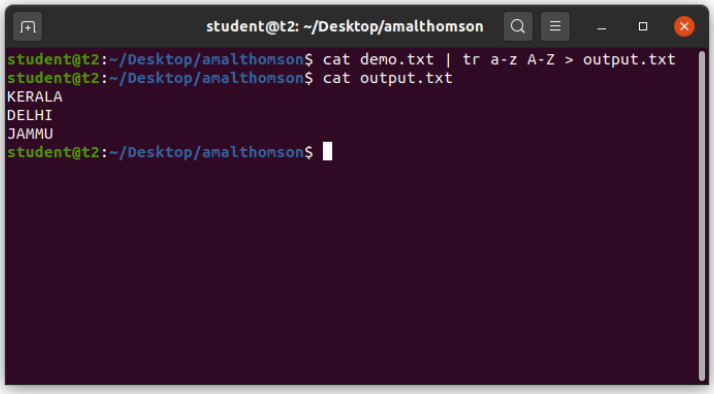
d) **cat -n** – to display the contents with line numbers.
Syntax: $ cat -n <*filename*>
Output:



e) **cat -b** – to remove numbering for empty lines.
Syntax: $ cat -n <*filename*>
Output:



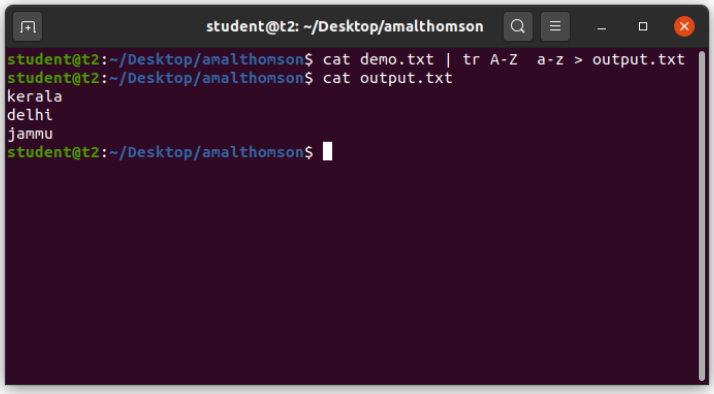
f) **cat <*filename*> | tr a-z A-Z > <*filename*>** – converts the contents of a file to UpperCase and saves into another file.
Syntax: $ cat <*filename*> | tr a-z A-Z > <*filename*>
Output:



a) **cat <*filename*> | tr A-Z a-z > <*filename*>** – converts the contents of a file to LowerCase and saves into another file.

Syntax: $ cat <*filename*> | tr A-Z a-z > <*filename*>

Output:



Result

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

Experiment No.: 4

Aim

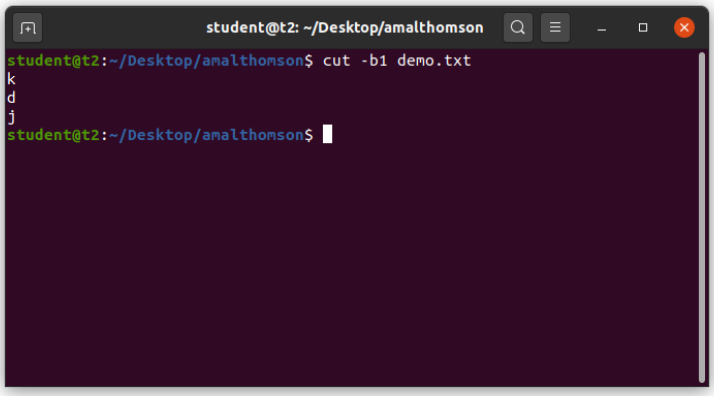
Familiarization of Linux Commands.

CO2

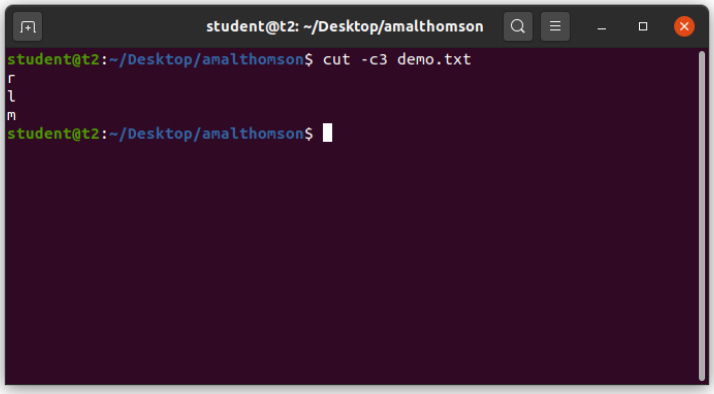
Perform system administration tasks.

Procedure

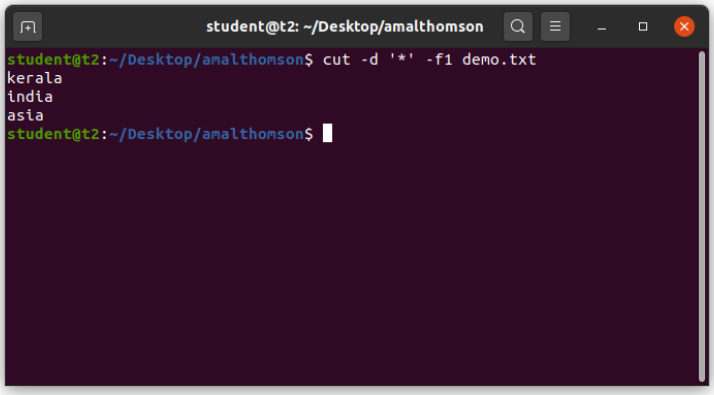
1. **cut** – to cut the contents of the file.
   Options of cut command:
   a) **cut -b1** – to cut the contents of a file by byte position.
   Syntax: $ cut -b1 *<filename>*
   Output:



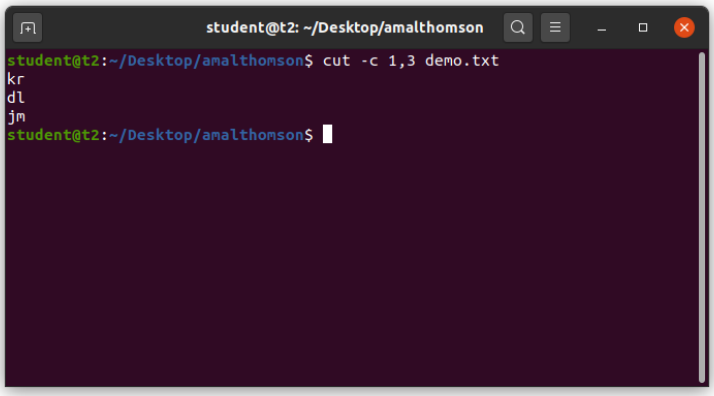
   b) **cut -c3** – to cut the contents of a file by character position.
   Syntax: $ cut -c3 *<filename>*
   Output:



   c) **cut –d '*' -f1** – use delimiter to cut the contents at '*' in the first column which is given by –f1.
   Syntax: $ cut –d '*' -f1 *<filename>*
   Output:

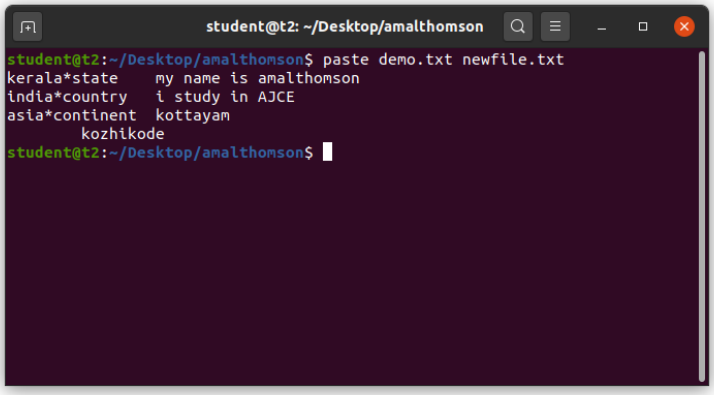


d) **cut –c** – to cut the characters from a specified position in a file.
Syntax: $ cut -c [1,3] *<filemane>*
Output:

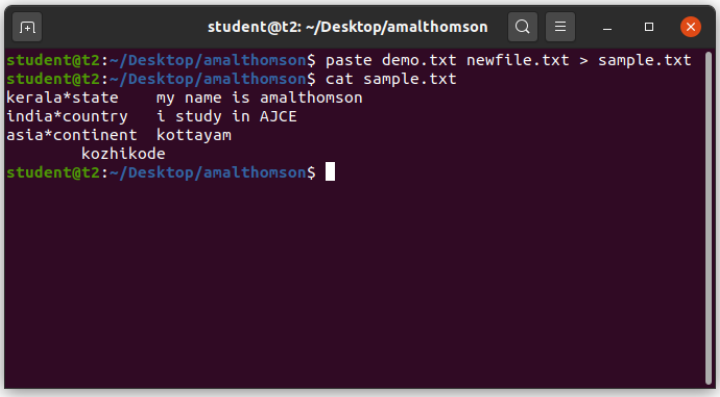


2. **paste** – to paste the content of a file to another.
Options of paste command

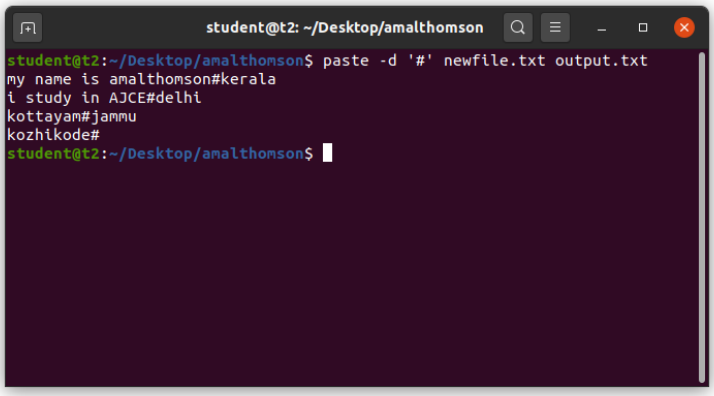
a) **paste *<filename> <filename>*** – to paste the contents in file1 to file2.
Syntax: $ paste *<filename> <filename>*
Output:



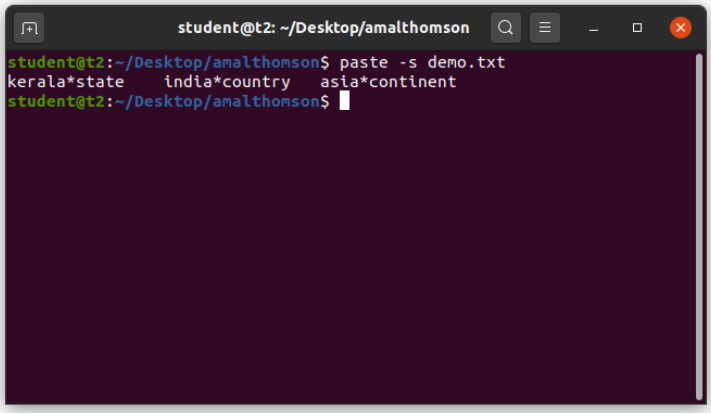
b) **paste *<filename> <filename> > <filename>*** – to paste the contents of two files to a third file.
Syntax: $ paste *<filename> <filename> > <filename>*
Output:



c) **paste –d '#' <filename> <filename>** – to paste # and join the contents of a file with another file.
Syntax: $ paste –d '#' *<filename> <filename>*
Output:



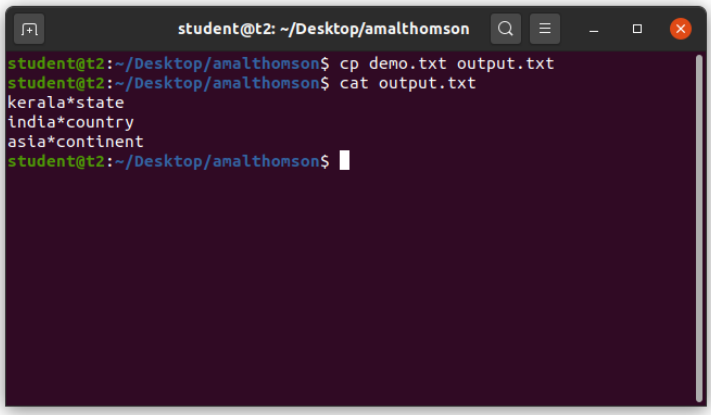
d) **paste -s** – to show all contents of a file in a single line.
Syntax: $ paste -s *<filename>*
Output:



3. **cp** – to copy the contents of a file.
Options of cp command:
   a) **cp *<filename> <filename>*** – to copy the contents of a file into another file or a new file.

Syntax: $ cp *<filename>* *<filename>*

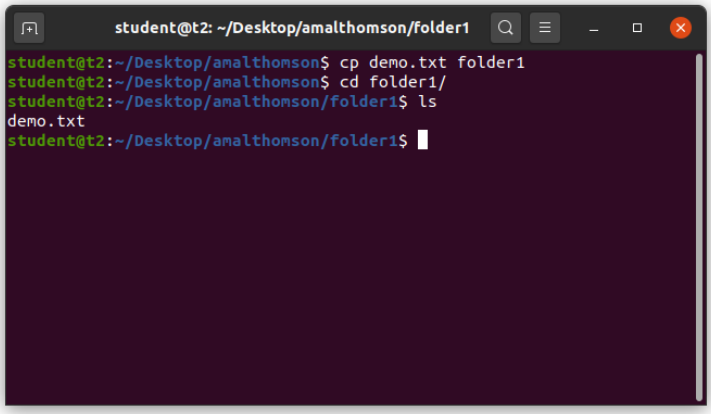
Output:



b) **cp *<filename>* *<directory>*** - to copy a file to a directory.

Syntax: $ cp *<filename>* *<directory>*

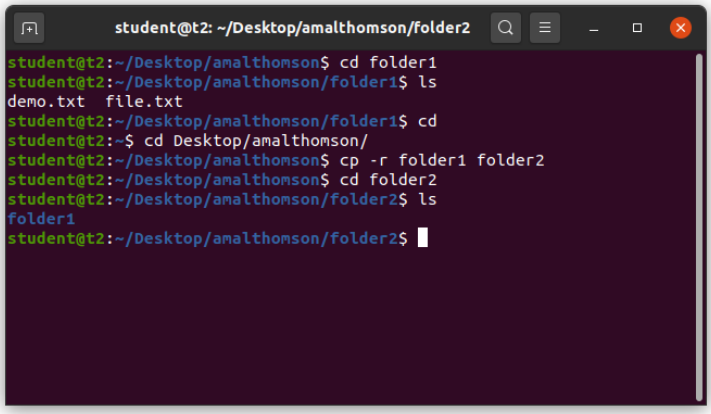
Output:



c) **cp -r** - to copy a directory and its contents to another directory.

Syntax: $ cp -r *<directory>* *<directory>*

Output:



Result

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

Experiment No.: 5

Aim

Familiarization of Linux Commands.

CO2

Perform system administration tasks.

Procedure

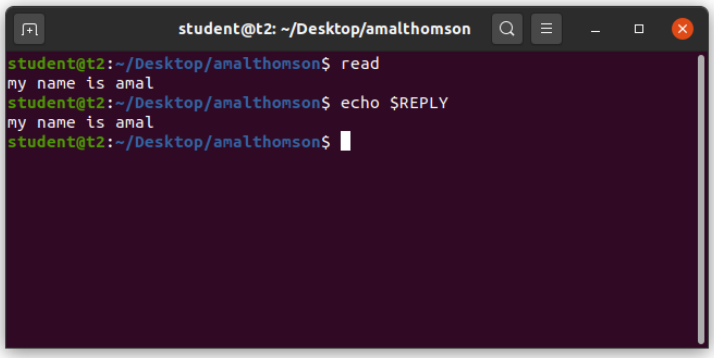
1. **read** – to read the contents of a line into a variable.

   Options of read command

   a) **read** – read contents of a line into variable.

   Syntax: $ read
   My name is amal
   $ echo $REPLY

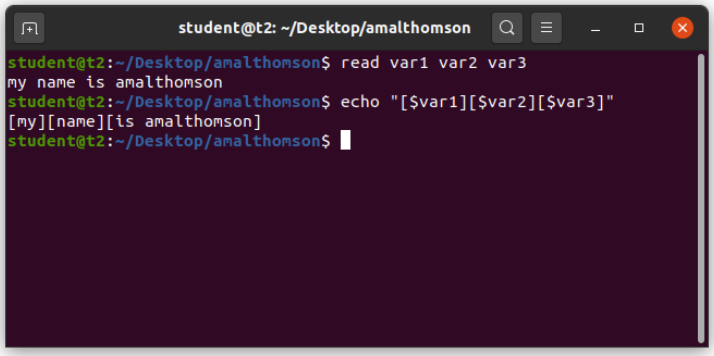
   Output:



   b) **read <*variable_name*>** – read contents of a line to a particular variables.

   Syntax: $ read var1 var2 var3
   My name is amalthomson
   $ echo "[var1][var2][var3]"

   Output:

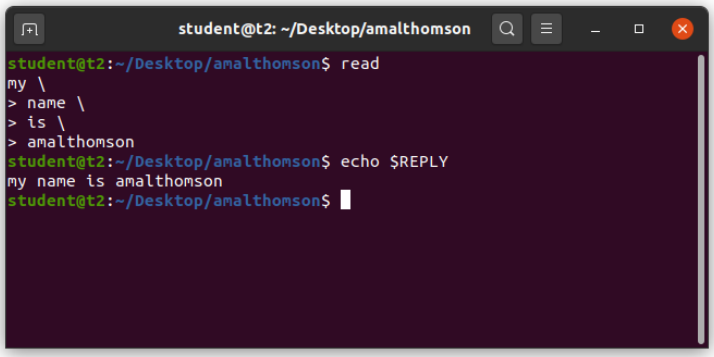


   c) **read** – read from multiple lines

   Syntax:

$ read
my \
name \
is \
amalthomson
$ echo $REPLY

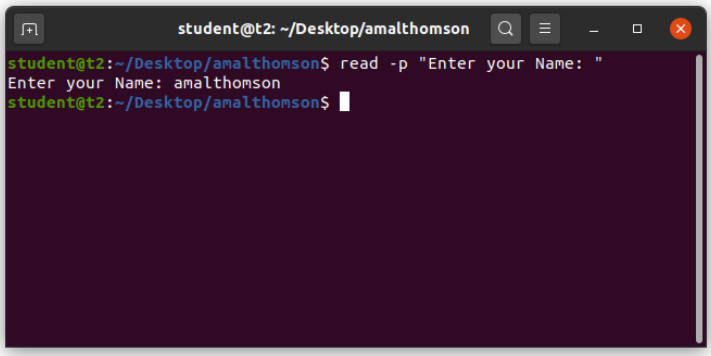
Outout:



d) **read -p** – read with prompt message

Syntax: $ read -p "Enter your name"

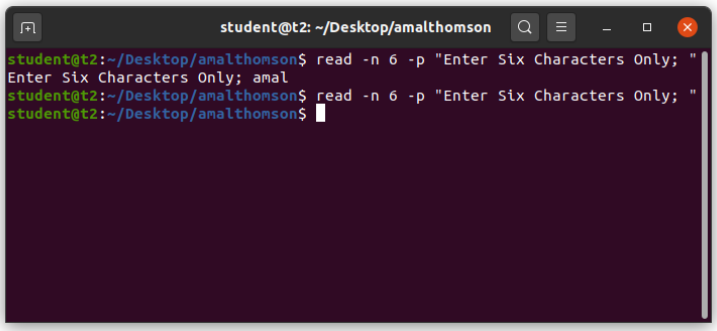
Output:



e) **read -n** – read with limit of characters can be read

Syntax: $ read -n -p "Enter only six characters"

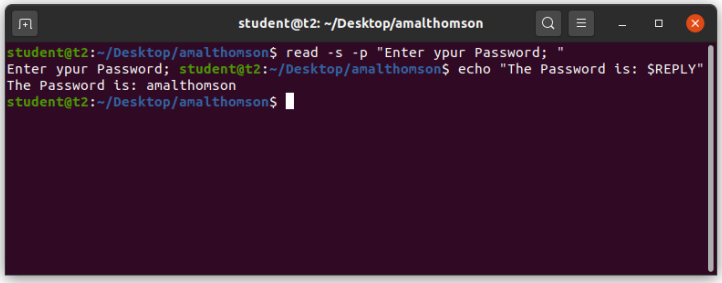
Output:



f) **read -s** – read lines securely without displaying the data entered

Syntax: $ read -s -p "Enter your password"

Output:



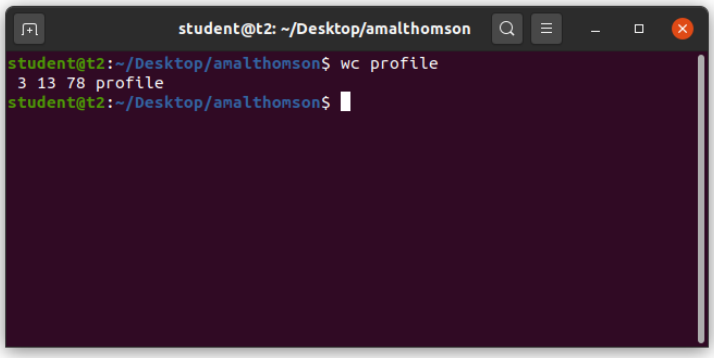
2. **wc** – word count

Options of wc commands

a) **wc <*filename*>** – to display number of lines, words, bytes and filename from a file

Syntax: $ wc profile.txt

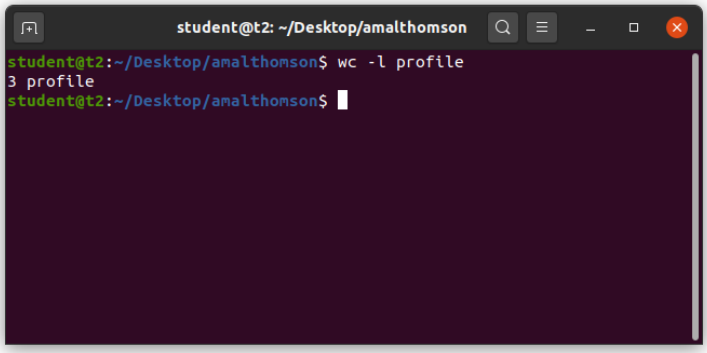
Output:



b) **wc -l <*filename*>** – to display number of lines and filename from a file

Syntax: $ wc -l profile.txt

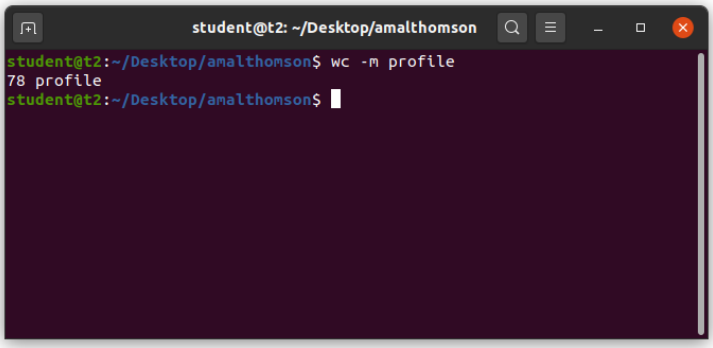
Output:



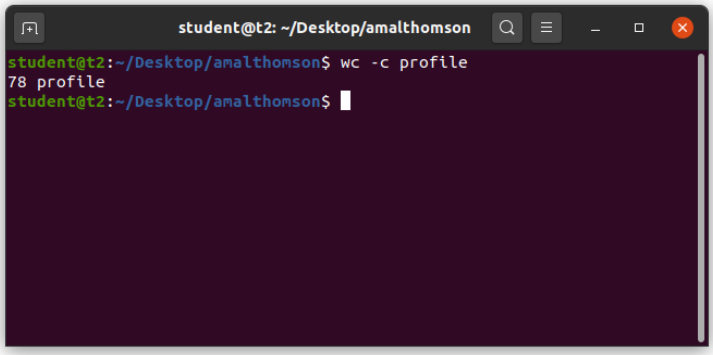
c) **wc -m <*filename*>** – to display number of bytes and filename from a file

Syntax: $ wc -m profile.txt

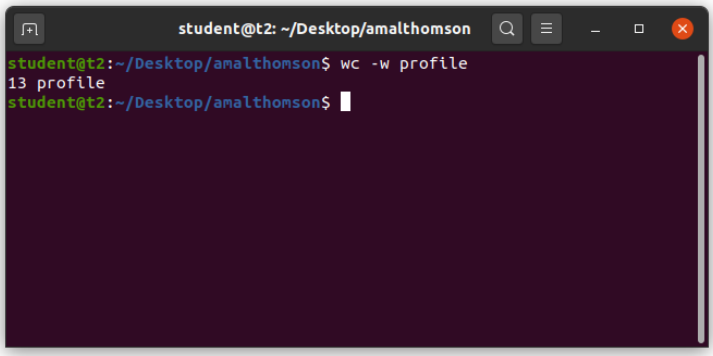
Output:



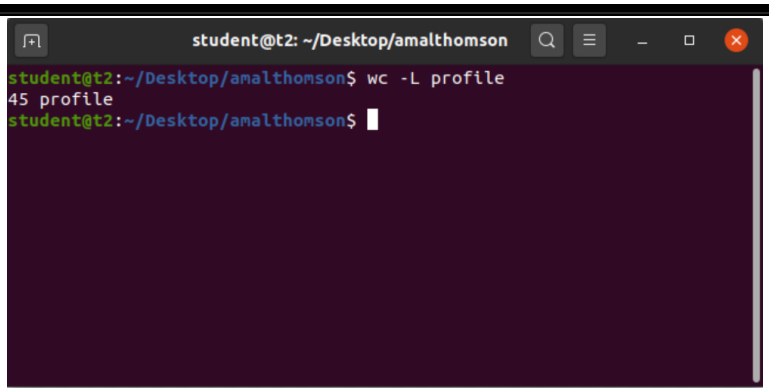
d) **wc -c <*filename*>** – to display number of characters and filename from a file
Syntax: $ wc -c profile.txt
Output:



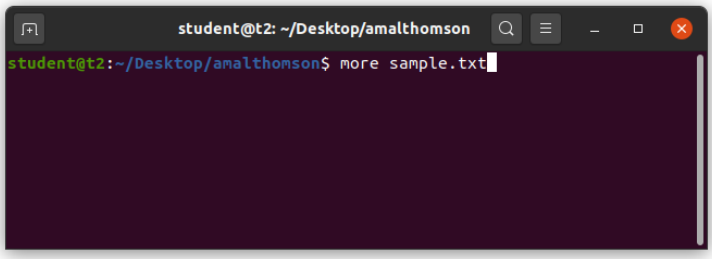
e) **wc -w <*filename*>** – to display number of words and filename from a file
Syntax: $ wc -w profile.txt
Output:

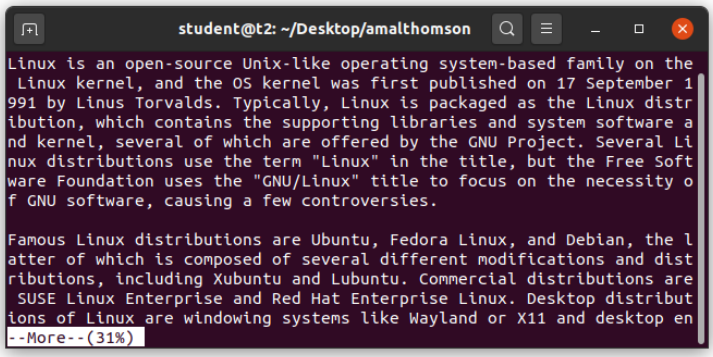


f) **wc -L <*filename*>** – to display length of largest line.
Syntax: $ wc -L profile.txt
Output:

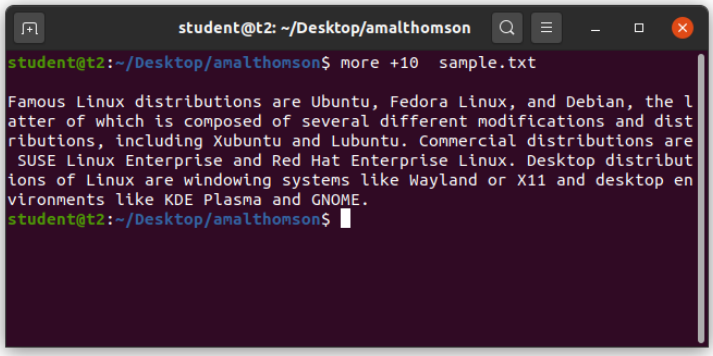


3. **more** – is similar to get to display the contents, the only difference is that in case of longer text or content get command output will scroll off your screen while more command display the output only screen full at a time.
   Options of more command
   a) **more <*filename*>** – display contents of a file
      Syntax: $ more sample.txt
      Output:





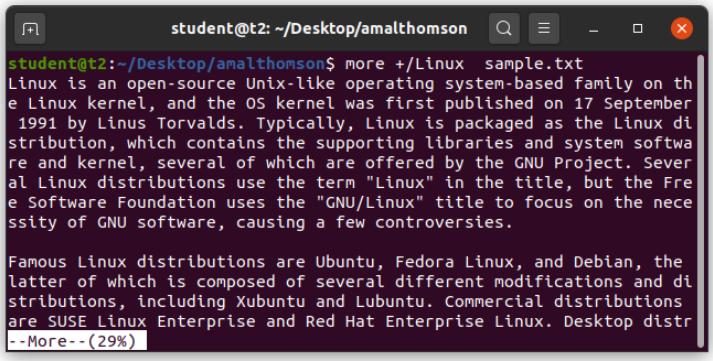
   b) **more +20 <*filename*>** – display contents of a file
      Syntax: $ more +10 sample.txt
      Output:



c) **more +/pattern <*filename*>** – to search is train inside your document, you can view all the instances by navigating through the result

Syntax: $ more +/Linux sample.txt

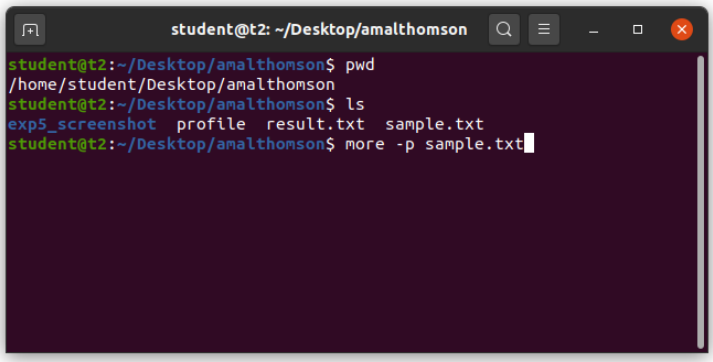
Output:

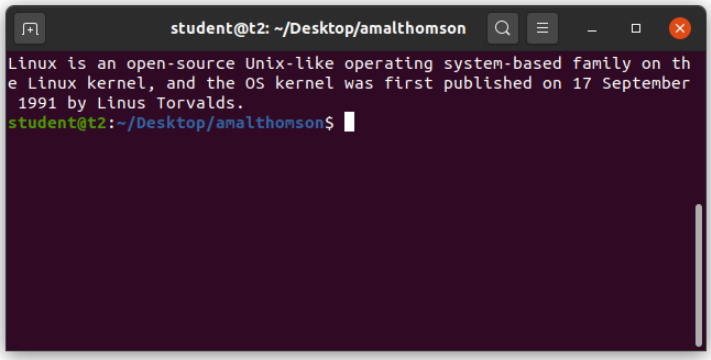


d) **more -p <*filename*>** – to display the contents of a file after clearing the screen

Syntax: $ more -p sample.txt

Output:

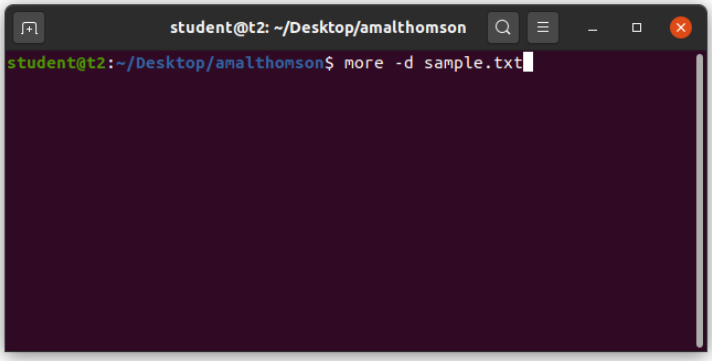


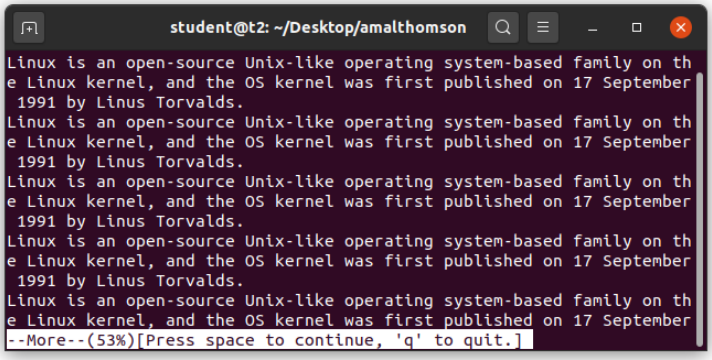


e) **more -d <*filename*>** – display instructions such as, space to continue and q to quit.

Syntax: $ more -d sample.txt

Output:





Result

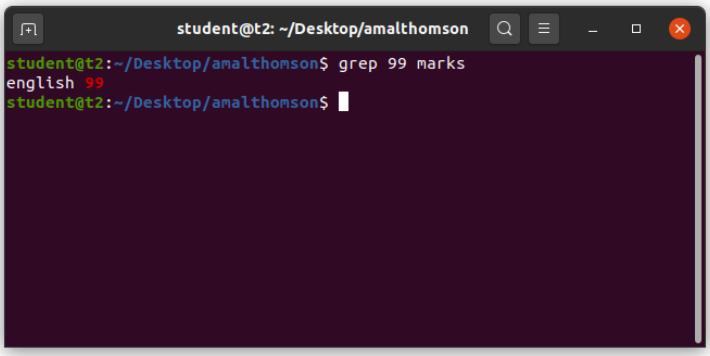
The program was executed and the result was successfully obtained. Thus CO2 was obtained.

Experiment No.: 7

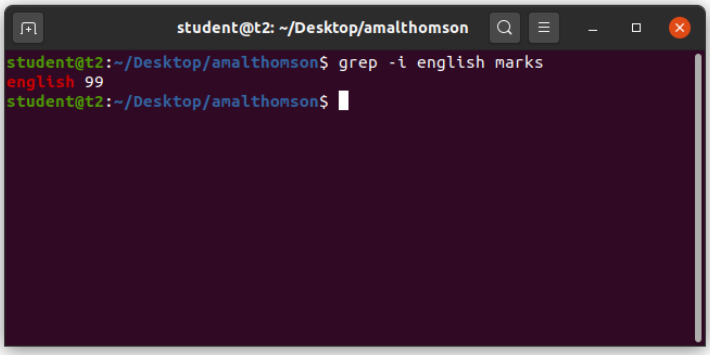
Aim
Familiarization of Linux Commands.

CO2
Perform system administration tasks.

Procedure
1. grep – used to filter the contents of a file, which makes search easy.
   Options of grep command
   a) grep <content><filename> – search and display a particular content from a file
      Syntax:
      $ grep 90 marks
      Output:



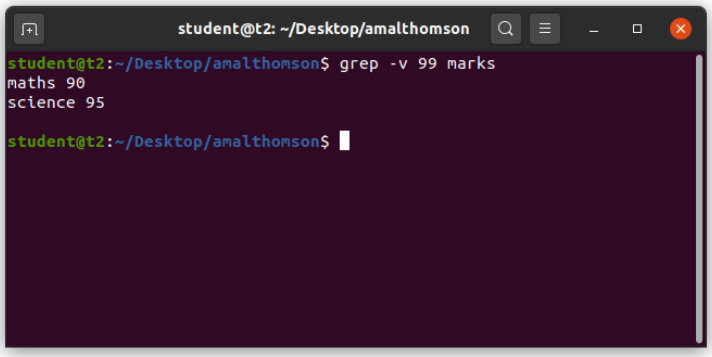
   b) grep -i <pattern><filename> – used to search and display a matching pattern, case insensitive
      Syntax:
      $ grep -i english marks
      Output:



   c) grep -v <content><filename> – inverted search and display
      Syntax:

$ grep -v 99 marks

Output:

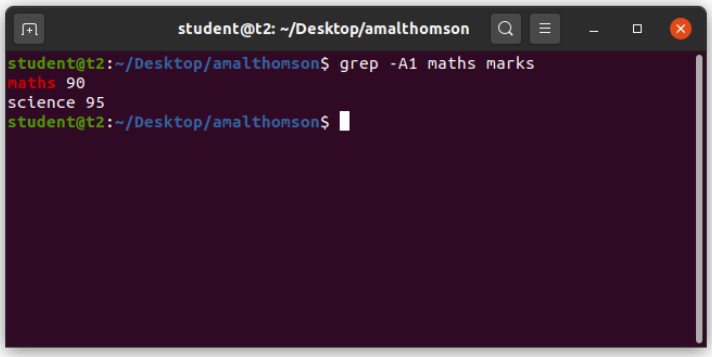


d) grep -A1 <content><filename> – display searched content and the next line from a file.

Syntax:

$ grep -A1 maths marks

Output:

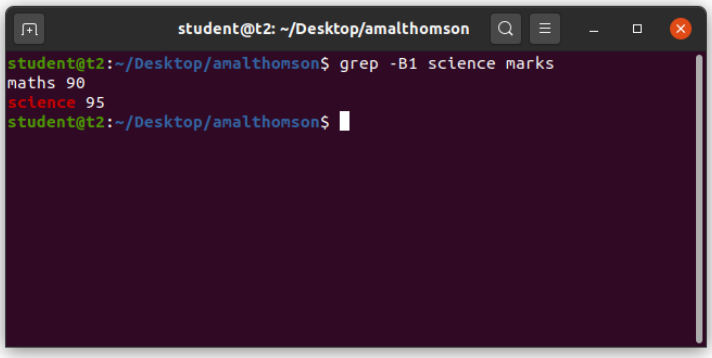


e) grep -B1 <content><filename> – display searched content and the previous line from a file.

Syntax:

$ grep -B1 science marks

Output:

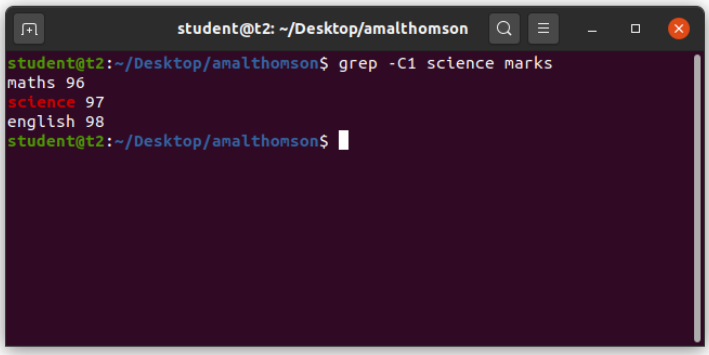


f) grep -C1 <content><filename> – display searched content and the previous and next line from a file.

Syntax:

$ grep -C1 science marks

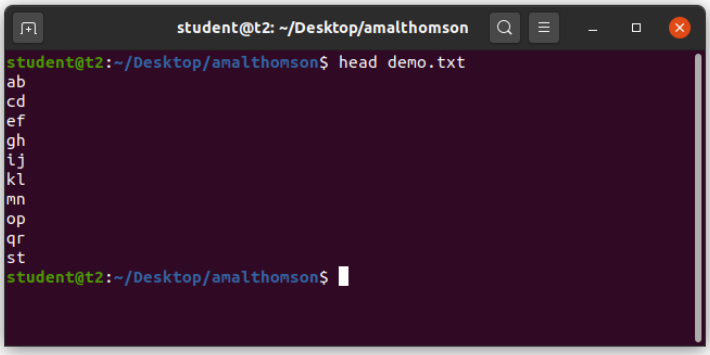
Output:



2. head – display top contents of the file, by default it displays top 10 lines.
   Options of head command.
   a) head <filename> – display top 10 lines of a file
      Syntax:

      $ head demo.txt

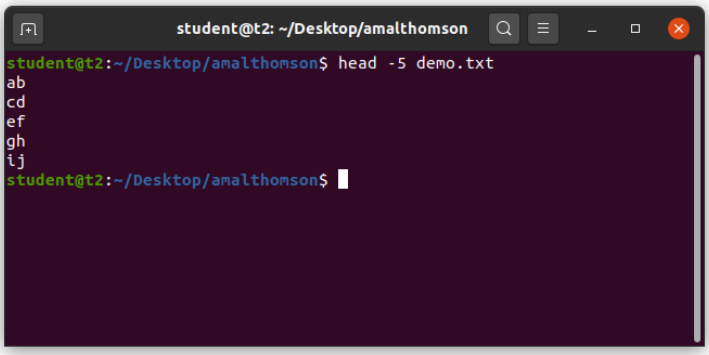
      Output:



   b) head -<limit> <filename> – display top number of lines mentioned in the limit of a file
      Syntax:

      $ head -5 demo.txt

      Output:



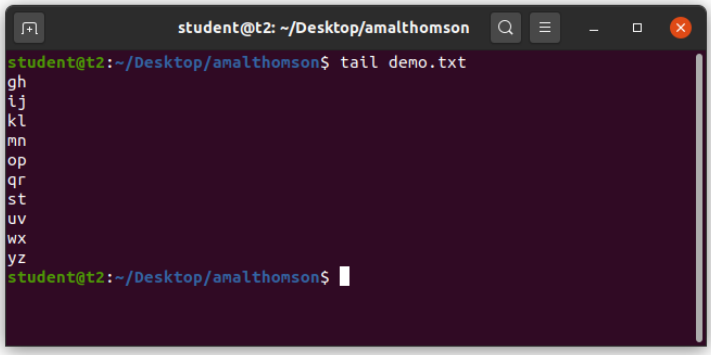
3. tail – display bottom contents of the file, by default it displays bottom 10 lines.
   Options of tail command.

a) tail <filename> – display bottom 10 lines of a file

Syntax:

$ tail demo.txt

Output:

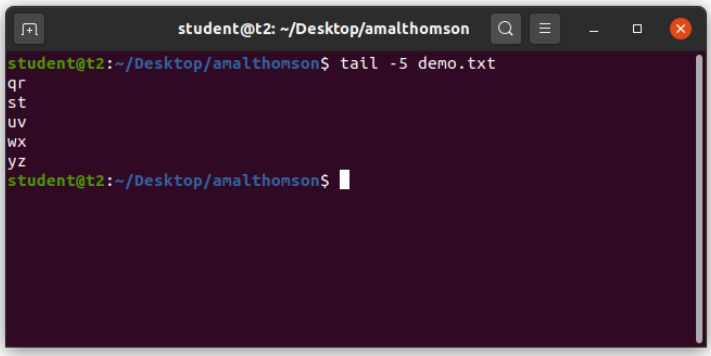


b) tail -<limit> <filename> – display bottom number of lines mentioned in the limit of a file

Syntax:

$ tail -5 demo.txt

Output:



4. mv – used to move files or folders.

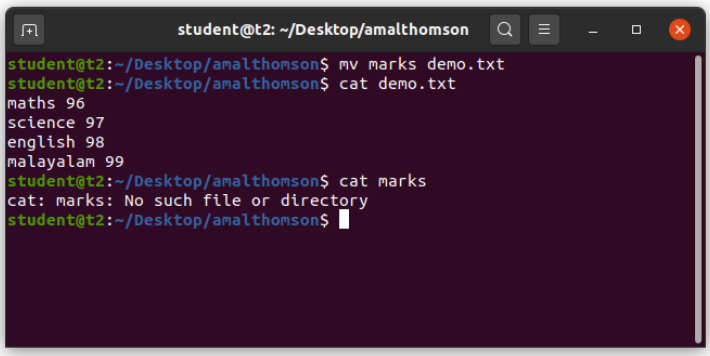
Options of move command

a) mv <filename> <filename> – replaces file2 with file1

Syntax:

$ mv marks demo.txt

Output:

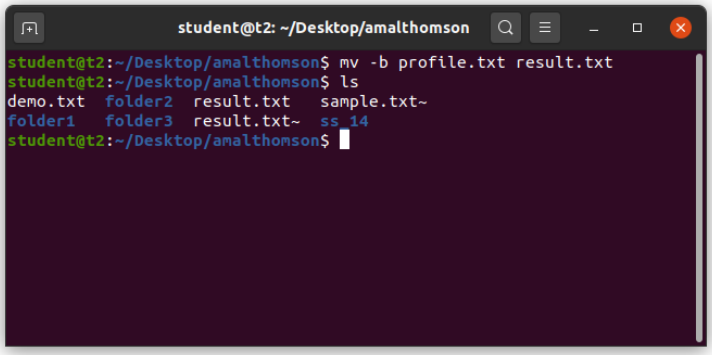


b) mv -b <filename> <filename> – replace file2 with file1 and keeps a backup of the file replaced

Syntax:

$ mv -b profile.txt result.txt

Output:

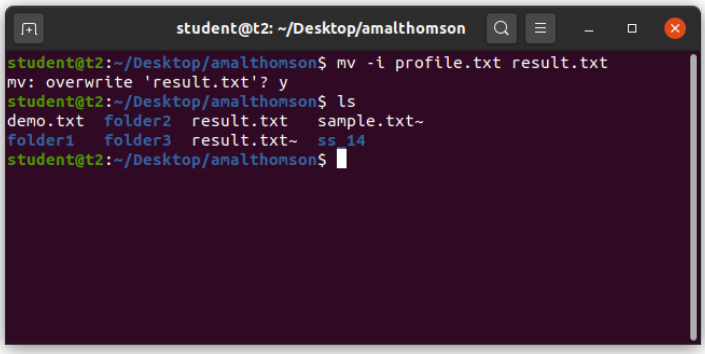


c) mv -i <filename> <filename> – displays a prompt message to confirm overwrite.

Syntax:

$ mv -i profile.txt result.txt

Output:



Result

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