**Experiment No.: 3**

**Aim**

Familiarization of Linux Commands.

**CO2**

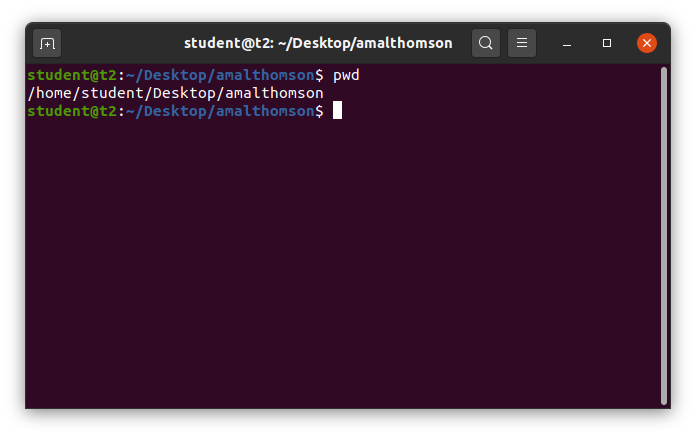
Perform system administration tasks.

**Procedure**

1. **pwd** – used to print the working directory. After execution it shows the absolute path.

Syntax: $ pwd

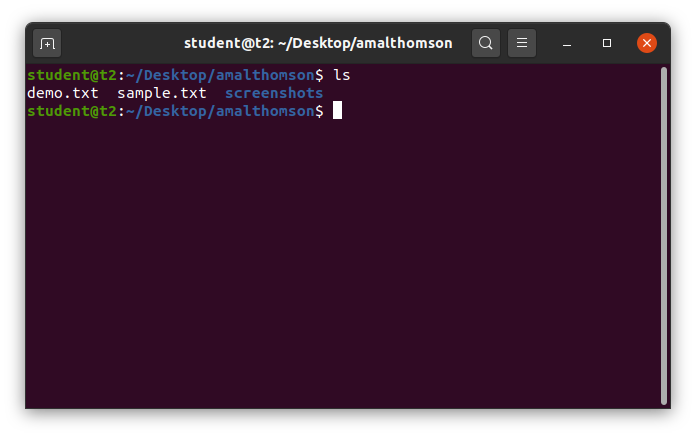
Output:



1. **ls** – used to list the files and content in the directory.

Syntax: $ ls

Output:

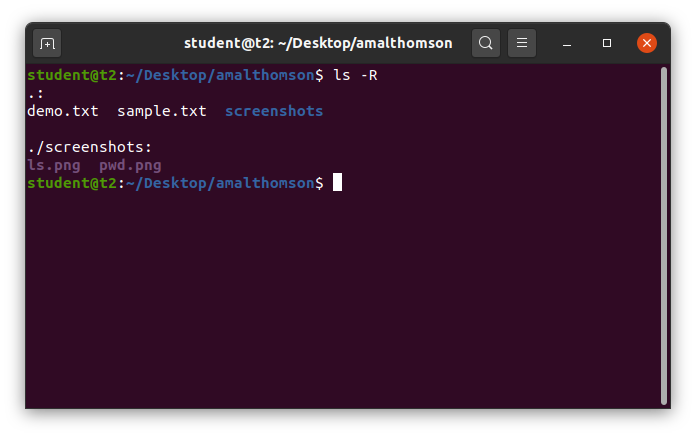


**Options of ls command.**

1. **ls -R** – used to list the directory as well as the subdirectory.

Syntax: $ ls -R

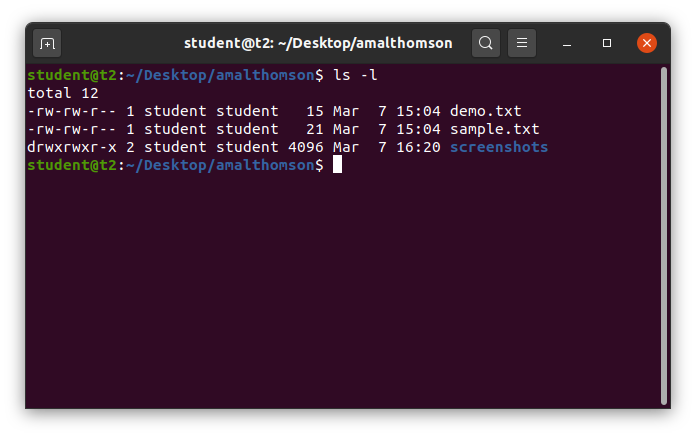
Output:



1. **ls -l** – used to view the long list of directory.

Syntax: $ ls -l

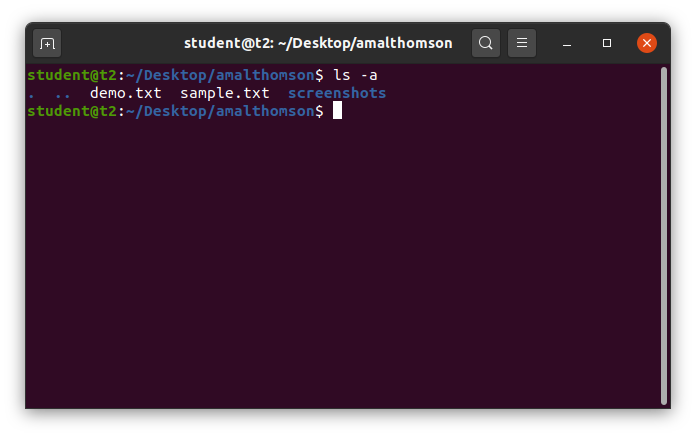
Output:



1. **ls -a** – used to view the list in directory along with hidden files.

Syntax: $ ls -a

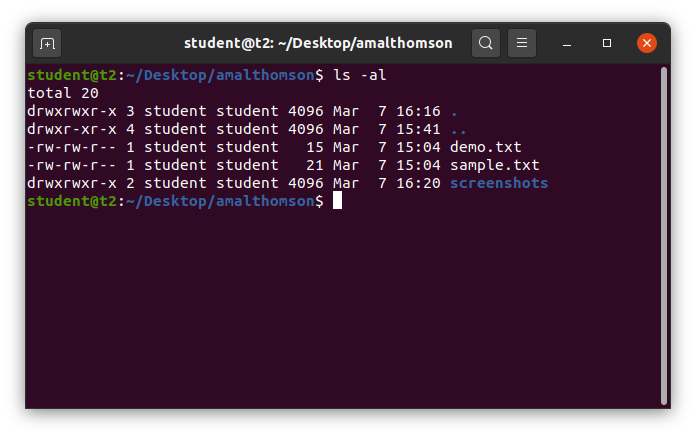
Output:



1. **ls -al** – used to view the list in directory with detailed information along with hidden files.

Syntax: $ ls -al

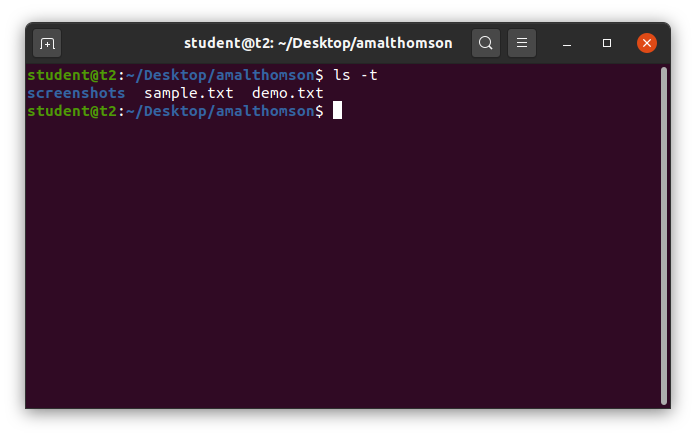
Output:



1. **ls -t** – used to view the list in sorted order of last modified.

Syntax: $ ls -t

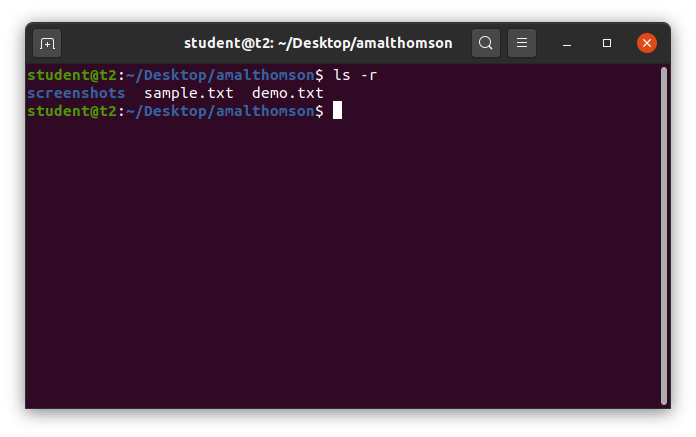
Output:



1. **ls -r** – used to view the list in reverse order of last modified.

Syntax: $ ls -r

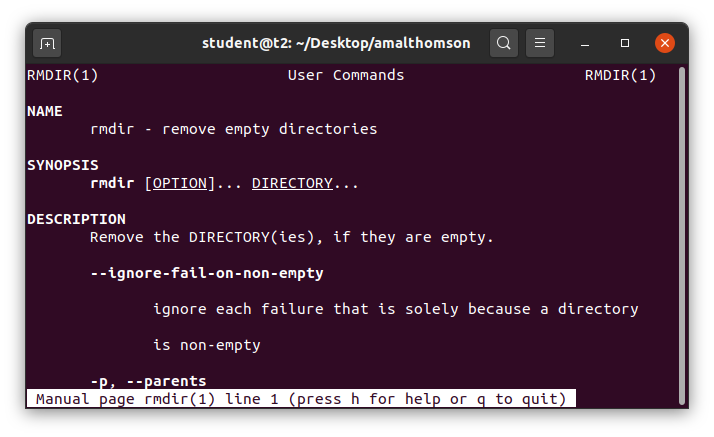
Output:



1. **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:



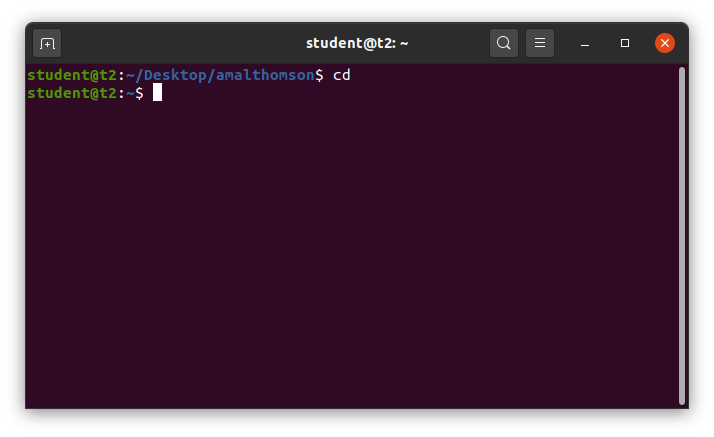
1. **cd** – used to navigate through directory.

**Options of cd commands**:

1. cd – used to switch to home directory.

Syntax: $ cd

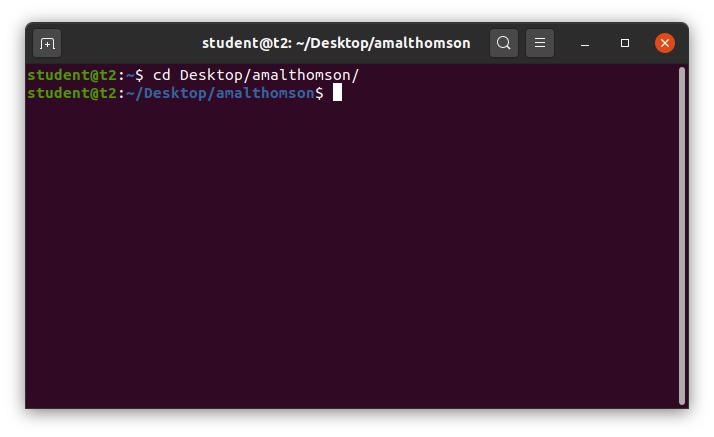
Output:



1. **cd <path>** - used to change to a particular path or directory

Syntax: $ cd *<directory\_path>*

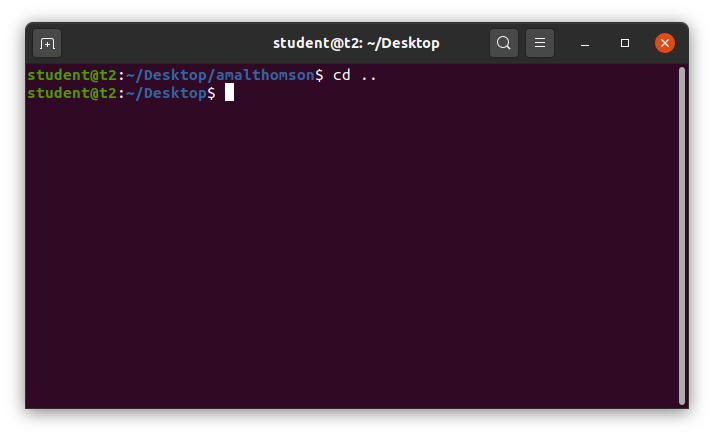
Output:



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

Syntax: $ cd ..

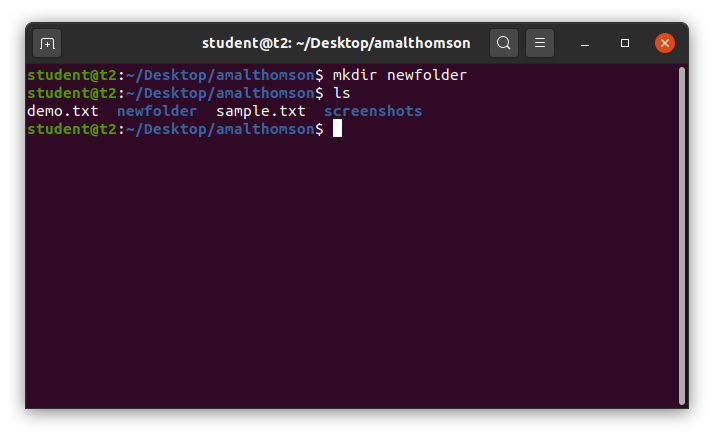
Output:



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

Syntax: $ mkdir *<directory\_name>*

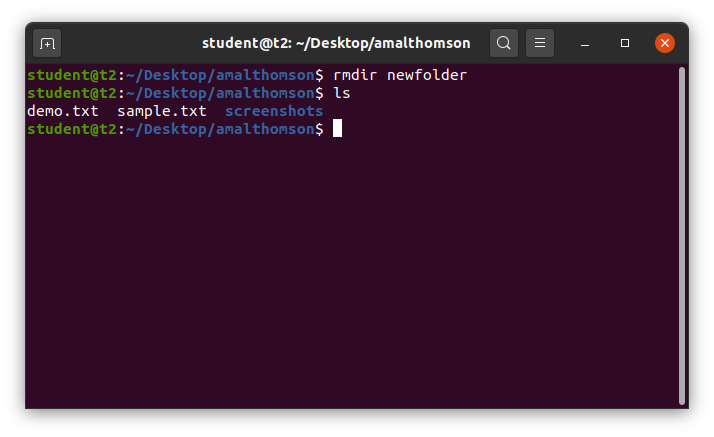
Output:



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

Syntax: $ rmdir <*directory\_name*>

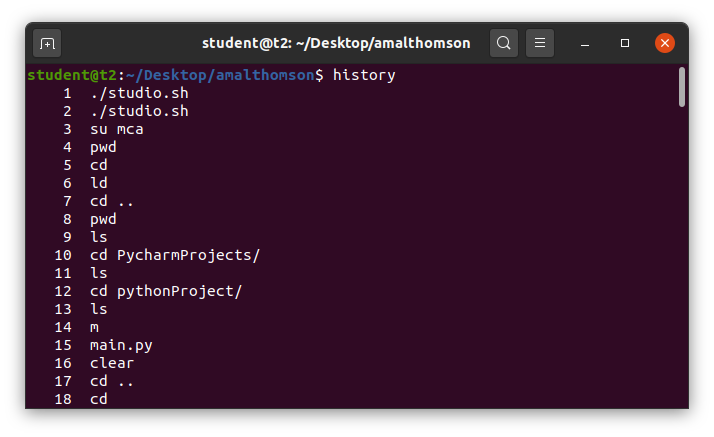
Output:



1. **history** – used to view the list of commands executed in a certain period of time.

Syntax: $ history

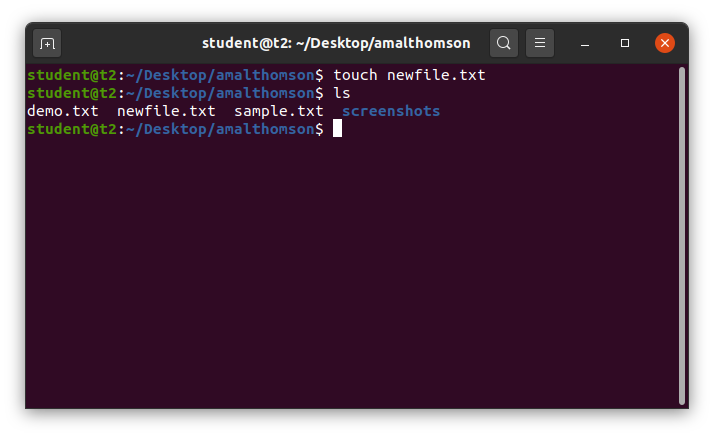
Output;



1. **touch** – used to create a new blank file.

Syntax: $ touch <*filename*>

Output:

****

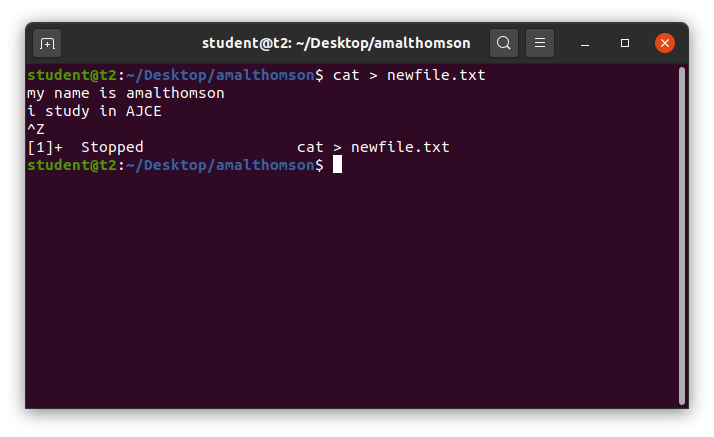
1. **cat –** usedto create a new blank file and also to add contents to the file.

**Options of cat commands:**

1. **cat >** – used to create a new blank file and also to add contents to the file.

Syntax: $ cat > <*filename*>

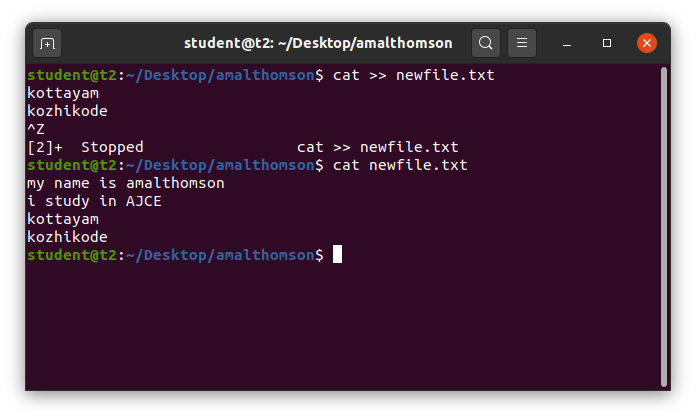
Output:



1. **cat >>** – used to append new contents to existing file.

Syntax: $ cat >> <*filename*>

Output:



1. **cat file1 file2 > file3** – copy contents of two files to a third file.

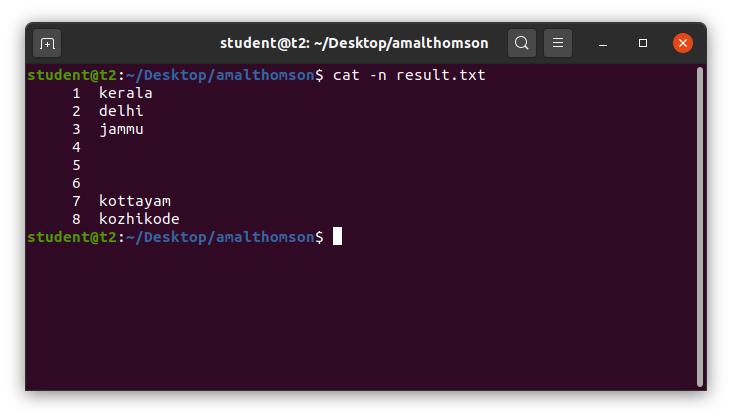
Syntax: $ cat *<filename><filename> > <filename>*

Output:

1. **cat -n** – to display the contents with line numbers.

Syntax: $ cat -n <*filename*>

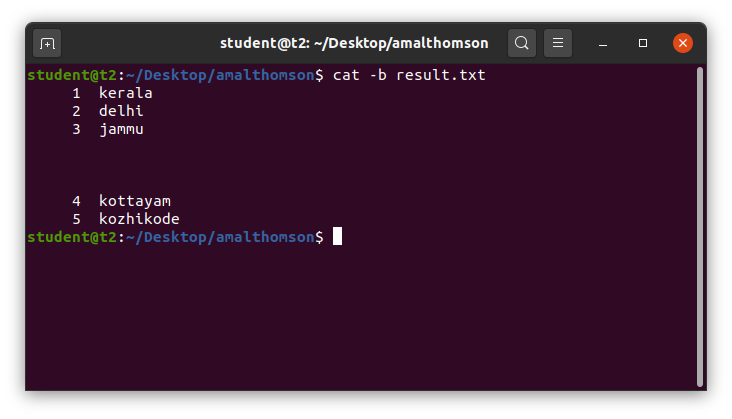
Output:



1. **cat -b** – to remove numbering for empty lines.

Syntax: $ cat -n <*filename>*

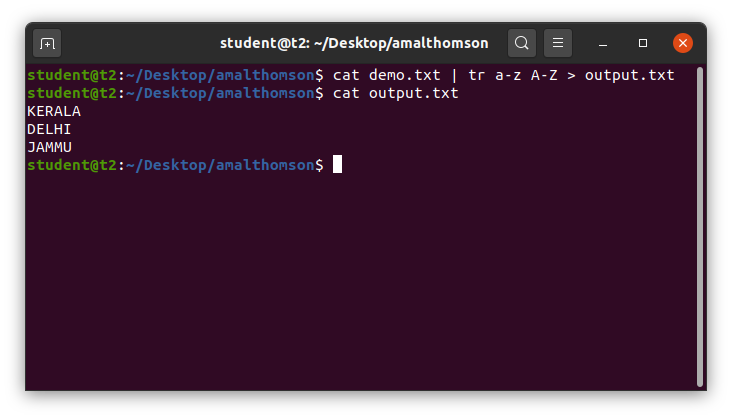
Output:



1. **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*>

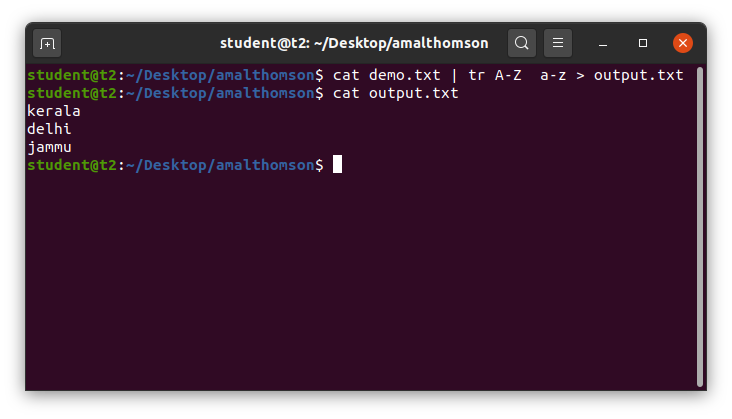
Output:



1. **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**

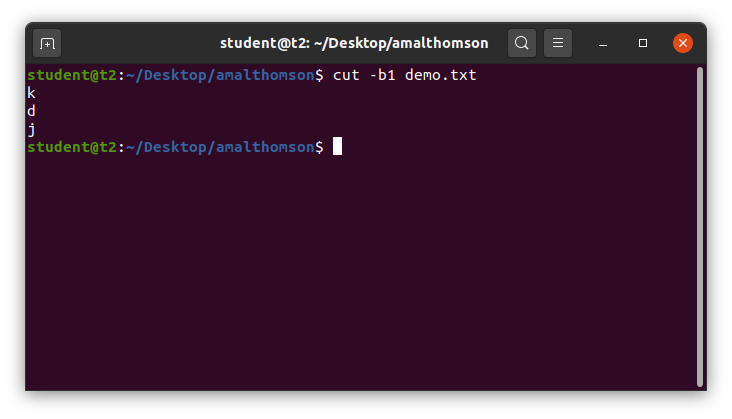
1. **cut** – to cut the contents of the file.

**Options of cut command**:

1. **cut -b1** – to cut the contents of a file by byte position.

Syntax: $ cut -b1 <*filename*>

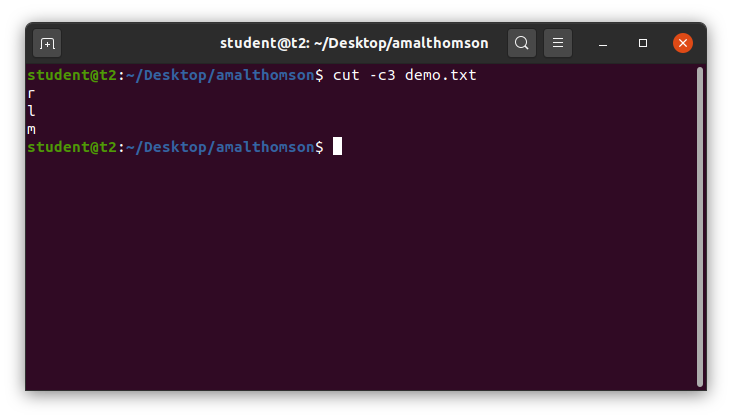
Output:



1. **cut -c3** – to cut the contents of a file by character position.

Syntax: $ cut -c3 <*filename*>

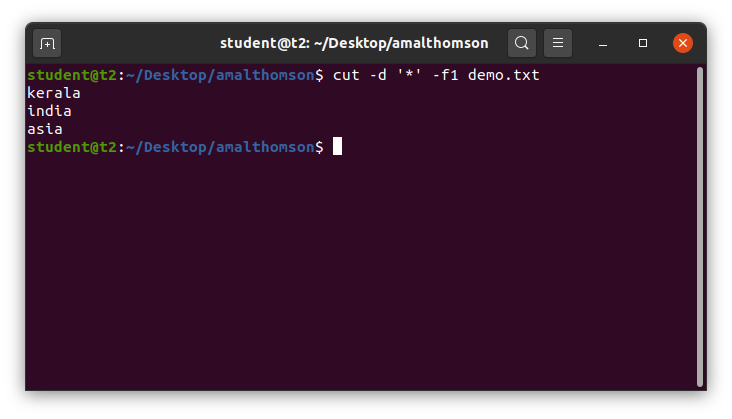
Output:



1. **cut –d ‘\*’ -f1** – use delimiter to cut the contents at ‘\*’ in the first column which is given by –f1.

Syntax: $ cut –d ‘\*’ -f1 <*filename*>

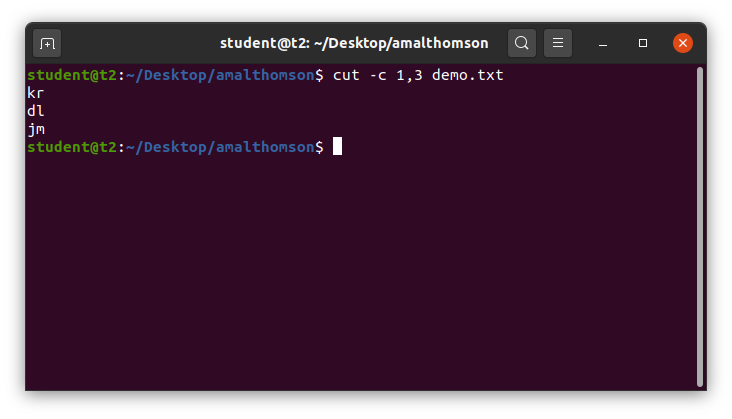
Output:



1. **cut –c** – to cut the characters from a specified position in a file.

Syntax: $ cut -c [1,3] <*filemane*>

Output:



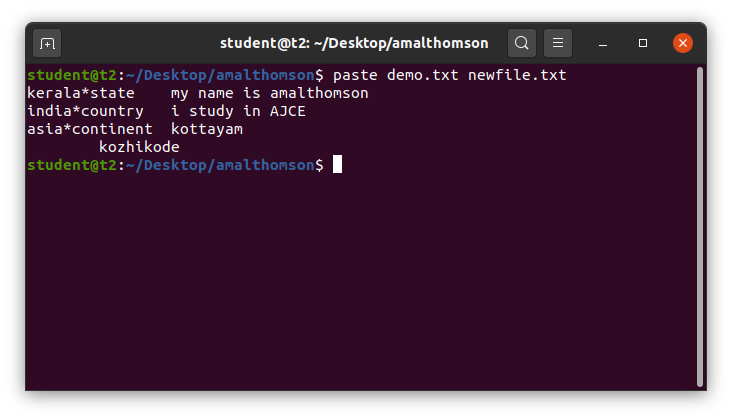
1. **paste** – to paste the content of a file to another.

**Options of paste command**

1. **paste <*filename*> <*filename*>** – to paste the contents in file1 to file2.

Syntax: $ paste <*filename*> <*filename*>

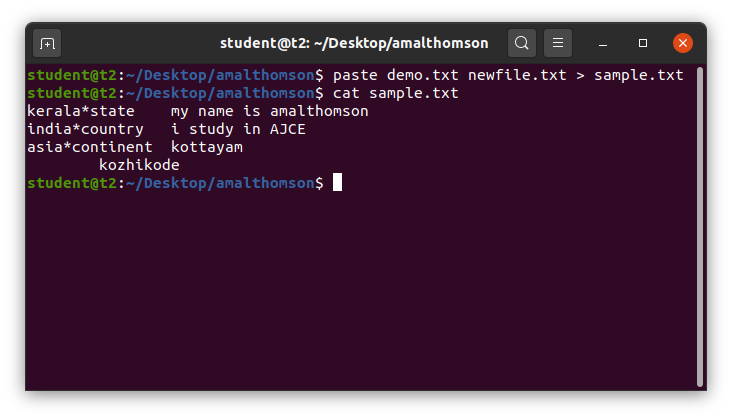
Output:



1. **paste *<filename> <filename> > <filename>*** *–* to paste the contents of two files to a third file.

Syntax: $ paste *<filename> <filename> > <filename>*

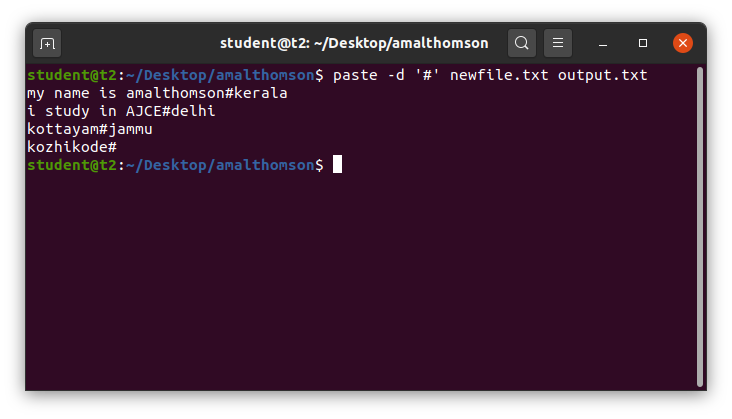
Output:



1. **paste –d ‘#’ <filename> <filename>** – to paste # and join the contents of a file with another file.

Syntax: $ paste –d ‘#’ *<filename> <filename>*

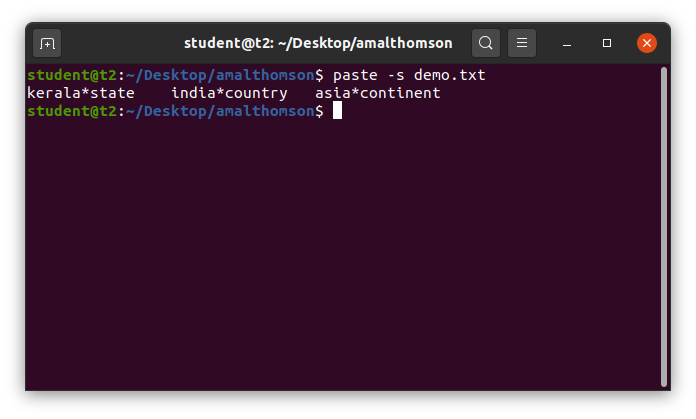
Output:



1. **paste -s** – to show all contents of a file in a single line.

Syntax: $ paste -s <*filename*>

Output:

****

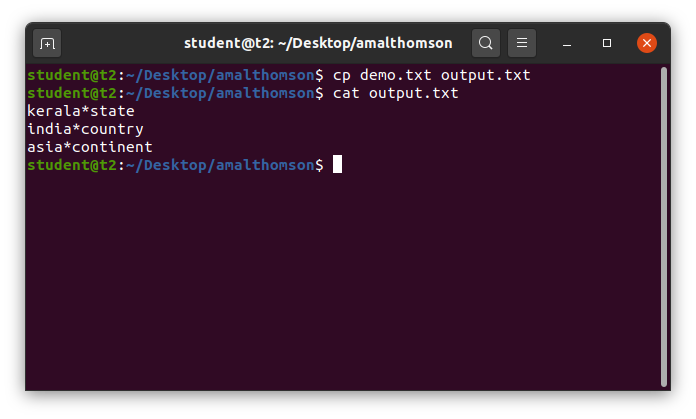
1. **cp** – to copy the contents of a file.

Options of cp command:

1. **cp *<filename> <filename>*** *–* to copy the contents of a file into another file or a new file.

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

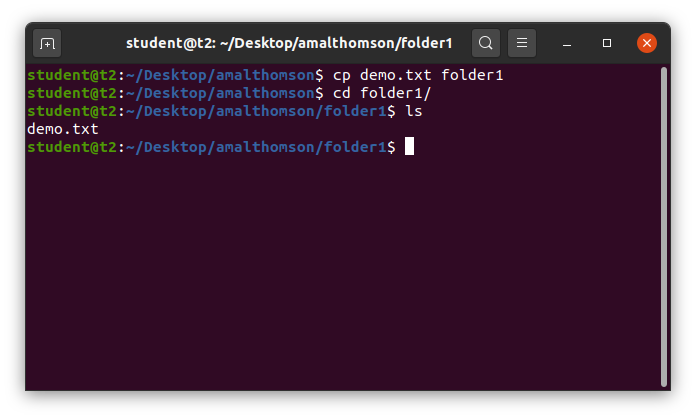
Output:



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

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

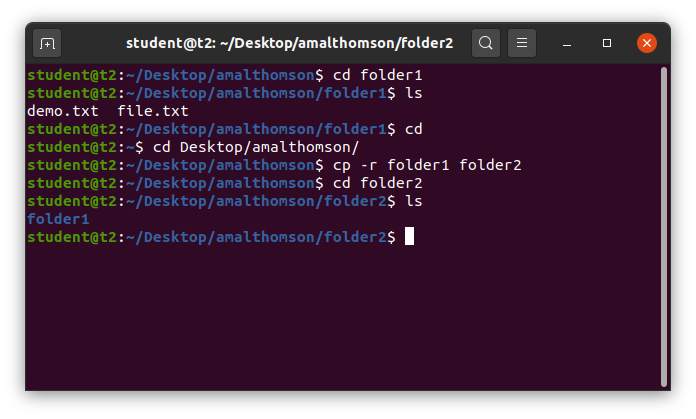
Output:



1. **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**

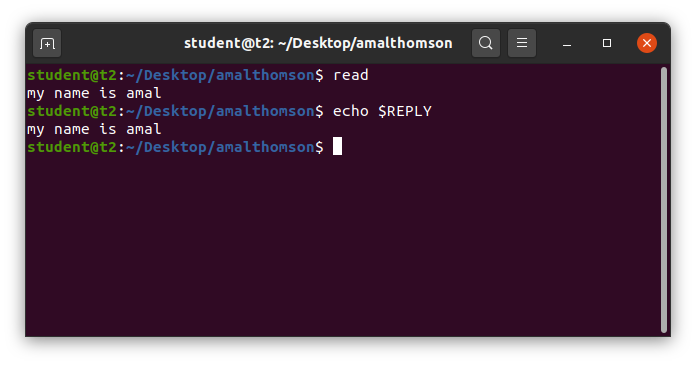
1. **read** – read contents of a line into variable.

Syntax: $ read

My name is amal

$ echo $REPLY

Output:



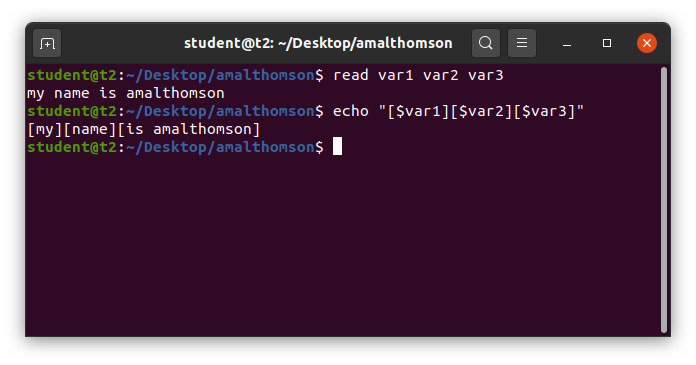
1. **read <*variable\_name*>** – read contents of a line to a particular variables.

Syntax: $ read var1 var2 var3

My name is amalthomson

$ echo “[var1][var2][var3]”

Output:



1. **read** – read from multiple lines

Syntax:

$ read

my \

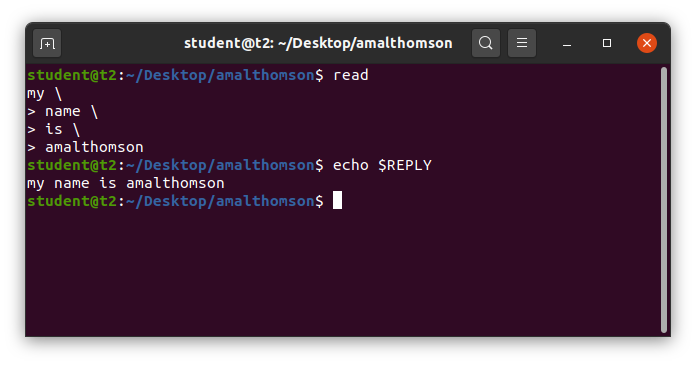
name \

is \

amalthomson

$ echo $REPLY

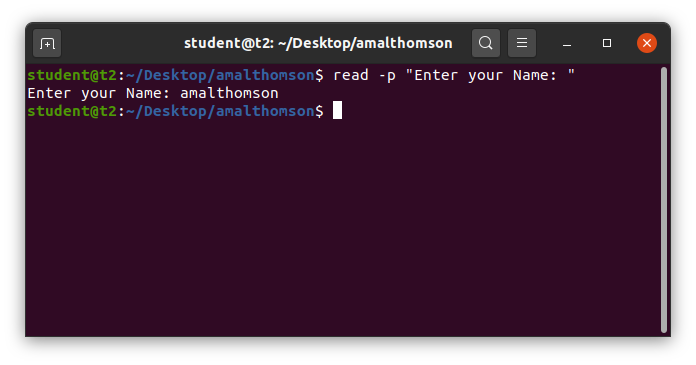
Outout:



1. **read -p** – read with prompt message

Syntax: $ read -p “Enter your name”

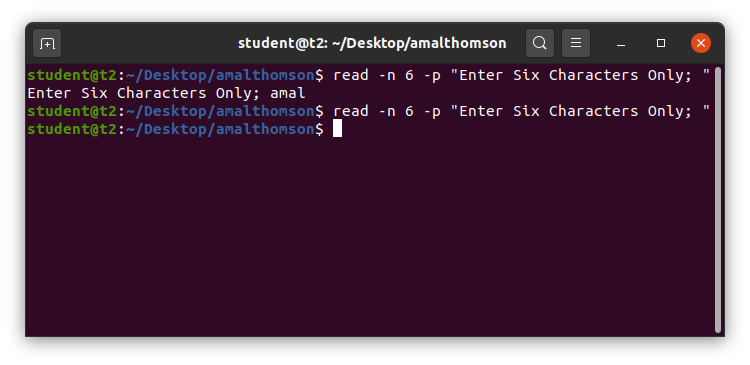
Output:



1. **read -n** – read with limit of characters can be read

Syntax: $ read -n -p “Enter only six characters”

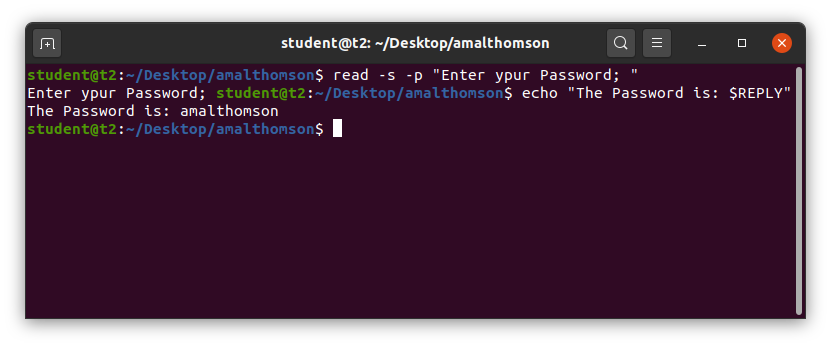
Output:



1. **read -s** – read lines securely without displaying the data entered

Syntax: $ read -s -p “Enter your password”

Output:



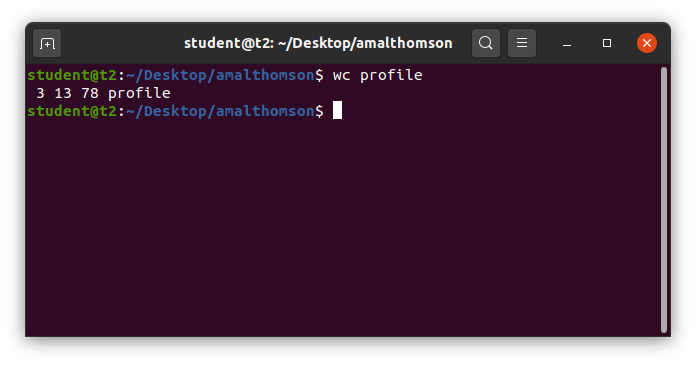
1. **wc** – word count

**Options of wc commands**

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

Syntax: $ wc profile.txt

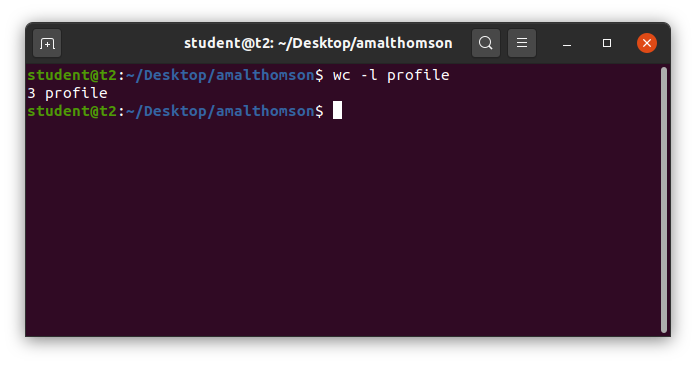
Output:

****

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

Syntax: $ wc -l profile.txt

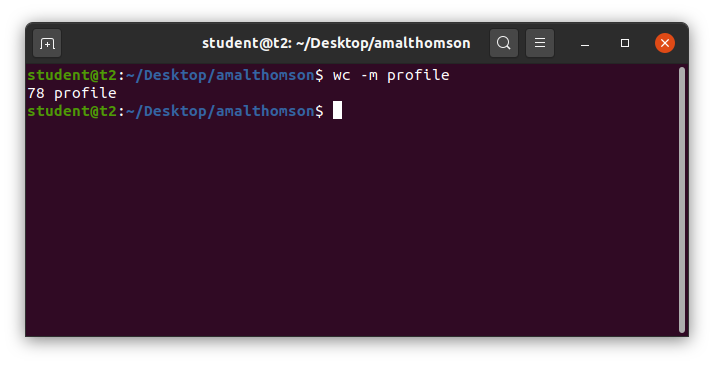
Output:



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

Syntax: $ wc -m profile.txt

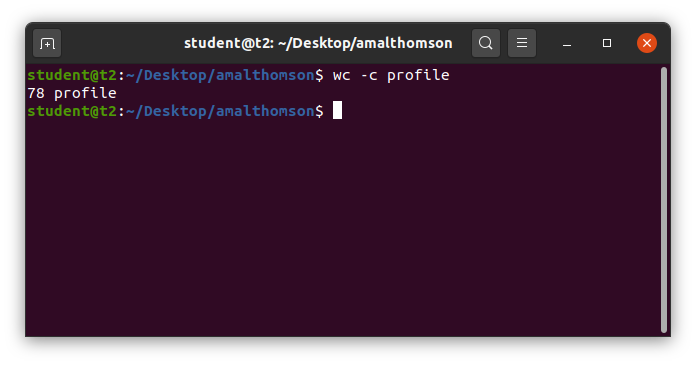
Output:



1. **wc -c <*filename*>** – to display number of characters and filename from a file

Syntax: $ wc -c profile.txt

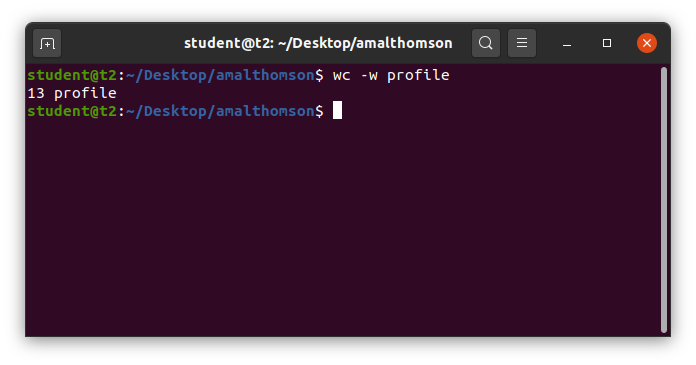
Output:



1. **wc -w <*filename*>** – to display number of words and filename from a file

Syntax: $ wc -w profile.txt

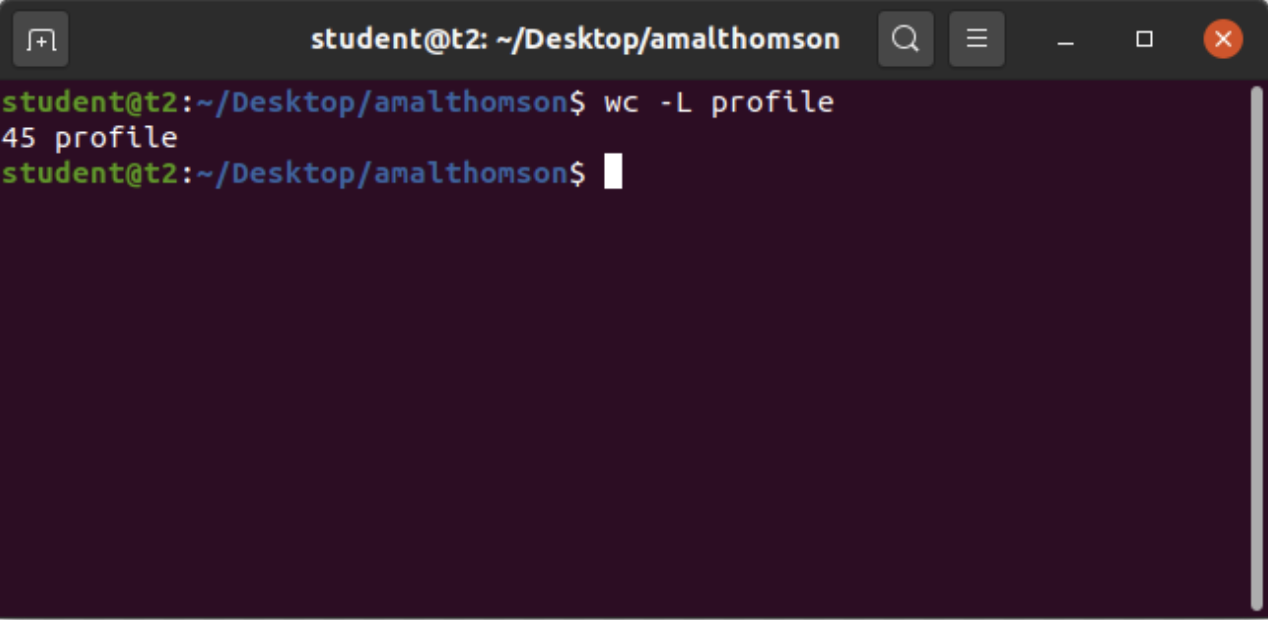
Output:



1. **wc -L <*filename***> – to display length of largest line.

Syntax: $ wc -L profile.txt

Output:



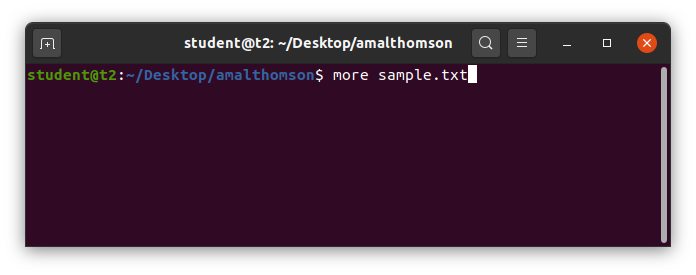
1. **more** – is similar to get to display the contents, the only difference is that in case of longer text or content get command output will scroll off your screen while more command display the output only screen full at a time.

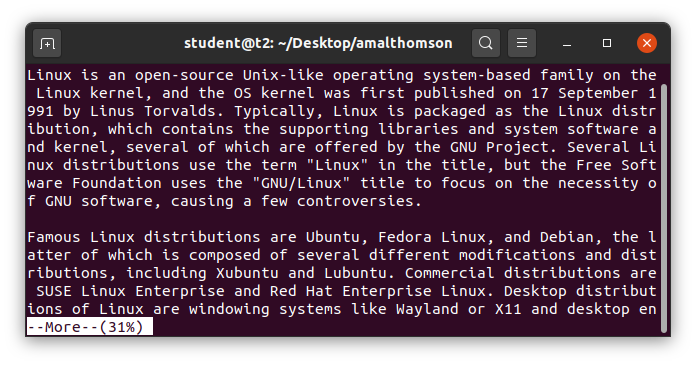
Options of more command

1. **more <*filename*>** – display contents of a file

Syntax: $ more sample.txt

Output:

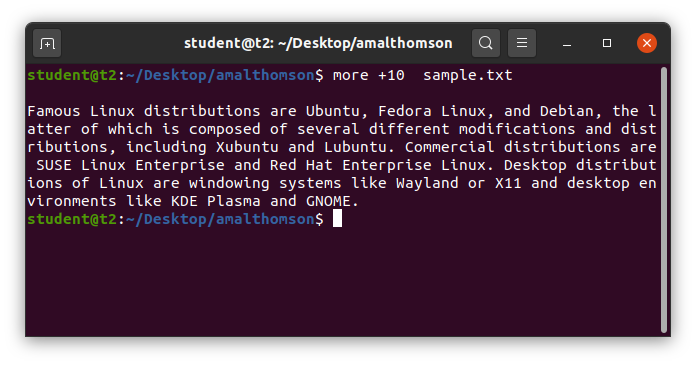




1. **more +20 <*filename*>** – display contents of a file

Syntax: $ more +10 sample.txt

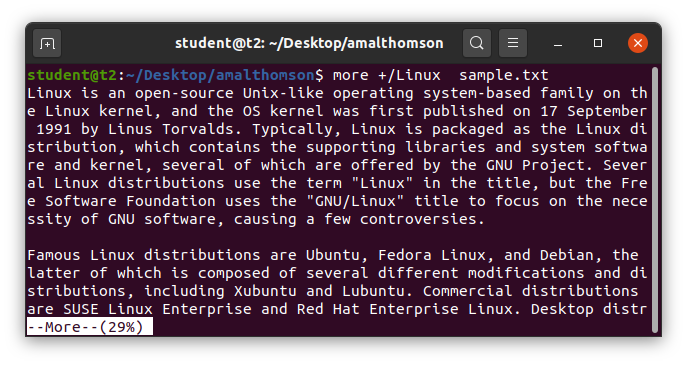
Output:



1. **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

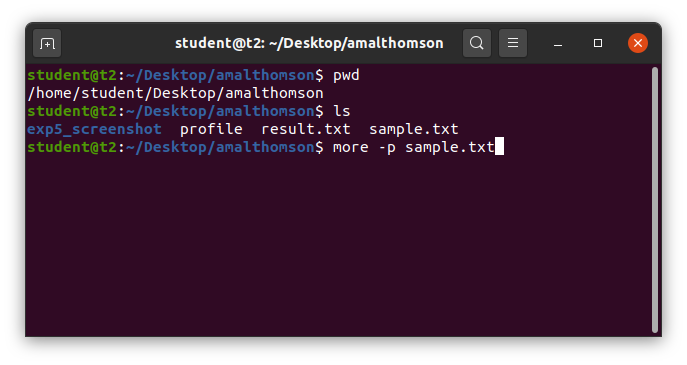
Output:

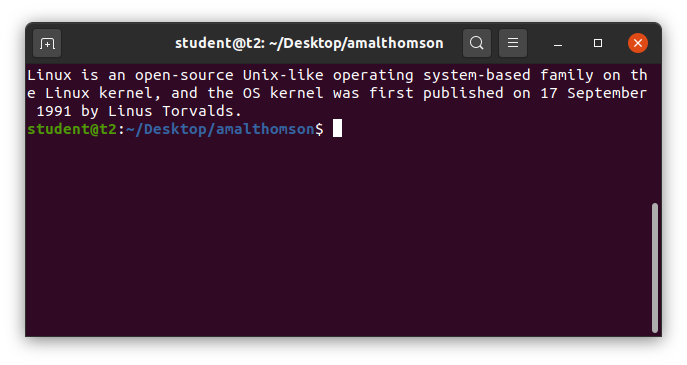


1. **more -p <*filename*>** – to display the contents of a file after clearing the screen

Syntax: $ more -p sample.txt

Output:

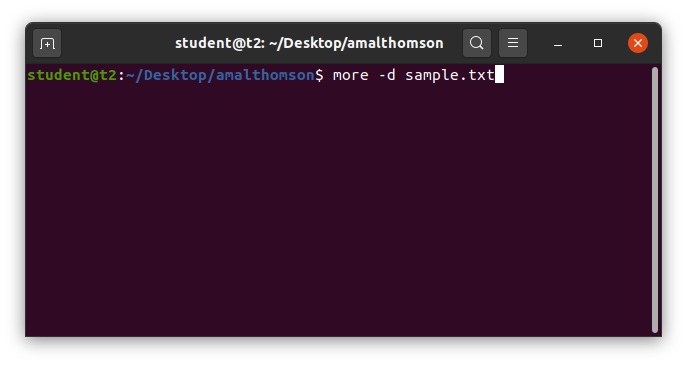


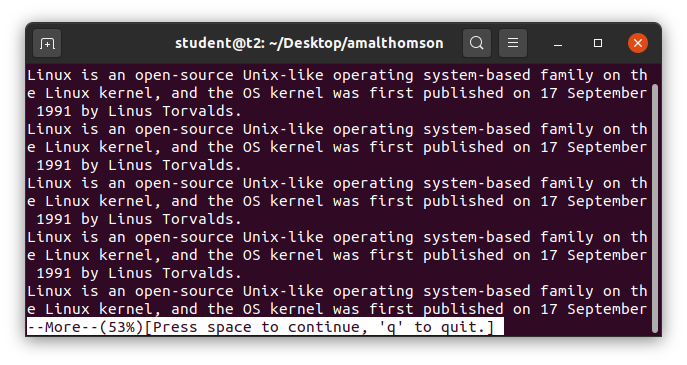


1. **more -d <*filename***> – display instructions such as, space to continue and q to quit.

Syntax: $ more -d sample.txt

Output:





**Result**

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