

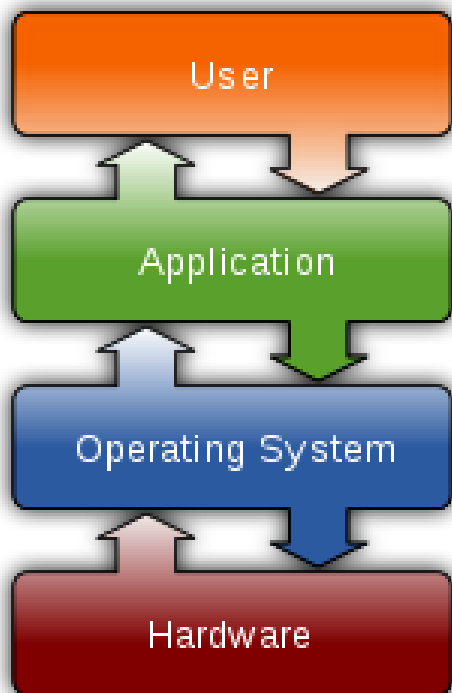
Practical 1

OPERATING SYSTEM

An **operating system (OS)** is system software that manages computer hardware and software resources and provides common services for computer programs.

For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware, although the application code is usually executed directly by the hardware and frequently makes system calls to an OS function or is interrupted by it. Operating systems are found on many devices that contain a computer – from cellular phones and video game consoles to web servers and supercomputers.

The dominant desktop operating system is Microsoft Windows with a market share of around 82.74%. macOS by Apple Inc. is in second place (13.23%), and the varieties of Linux are collectively in third place (1.57%). In the mobile (smartphone and tablet combined) sector, use in 2017 is up to 70% of Google's Android and according to third quarter 2016 data, Android on smartphones is dominant with 87.5 percent and a growth rate 10.3 percent per year, followed by Apple's iOS with 12.1 percent and a per year decrease in market share of 5.2 percent, while other operating systems amount to just 0.3 percent. Linux distributions are dominant in the server and supercomputing sectors. Other specialized classes of operating systems, such as embedded and real-time systems, exist for many applications.



Types Of Operating System

Single- and multi-tasking

A single-tasking system can only run one program at a time, while a multi-tasking operating system allows more than one program to be running in concurrency. This is achieved by time-sharing, dividing the available processor time between multiple processes that are each interrupted repeatedly in time slices by a task-scheduling subsystem of the operating system. Multi-tasking may be characterized in preemptive and co-operative types. In preemptive multitasking, the operating system slices the CPU time and dedicates a slot to each of the programs. Unix-like operating systems, e.g., Solaris, Linux, as well as AmigaOS support preemptive multitasking. Cooperative multitasking is achieved by relying on each process to provide time to the other processes in a defined manner. 16-bit versions of Microsoft Windows used cooperative multi-tasking. 32-bit versions of both Windows NT and Win9x, used preemptive multi-tasking.

Single- and multi-user

Single-user operating systems have no facilities to distinguish users, but may allow multiple programs to run in tandem. A multi-user operating system extends the basic concept of multi-tasking with facilities that identify processes and resources, such as disk space, belonging to multiple users, and the system permits multiple users to interact with the system at the same time. Time-sharing operating systems schedule tasks for efficient use of the system and may also include accounting software for cost allocation of processor time, mass storage, printing, and other resources to multiple users.

Distributed

A distributed operating system manages a group of distinct computers and makes them appear to be a single computer. The development of networked computers that could be linked and communicate with each other gave rise to distributed computing. Distributed computations are carried out on more than one machine. When computers in a group work in cooperation, they form a distributed system.

Templated

In an OS, distributed and cloud computing context, templating refers to creating a single virtual machine image as a guest operating system, then saving it as a tool for multiple running virtual machines. The technique is used both in virtualization and cloud computing management, and is common in large server warehouses.

Embedded

Embedded operating systems are designed to be used in embedded computer systems. They are designed to operate on small machines like PDAs with less autonomy. They are able to operate with a limited number of resources. They are very compact and extremely efficient by design. Windows CE and Minix 3 are some examples of embedded operating systems.

Real-time

A real-time operating system is an operating system that guarantees to process events or data by a specific moment in time. A real-time operating system may be single- or multi-tasking, but when multitasking, it uses specialized scheduling algorithms so that a deterministic nature of behavior is achieved. An event-driven system switches between tasks based on their priorities or external events while time-sharing operating systems switch tasks based on clock interrupts.

Library

A library operating system is one in which the services that a typical operating system provides, such as networking, are provided in the form of libraries and composed with the application and configuration code to construct a unikernel: a specialized, single address space, machine image that can be deployed to cloud or embedded environments.

Some of the operating systems

UNIX

Unix is a family of multitasking, multiuser computer operating systems that derive from the original AT&T Unix, development starting in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others.

Initially intended for use inside the Bell System, AT&T licensed Unix to outside parties in the late 1970s, leading to a variety of both academic and commercial Unix variants from vendors like the University of California, Berkeley (BSD), Microsoft (Xenix), IBM (AIX), and Sun Microsystems (Solaris). In the early 1990s, AT&T sold its rights in Unix to Novell, which then sold its Unix business to the Santa Cruz Operation (SCO) in 1995. The UNIX trademark passed to The Open Group, a neutral industry consortium, which allows the use of the mark for certified operating systems that comply with the Single UNIX Specification (SUS). As of 2014, the Unix version with the largest installed base is Apple's macOS.

Microsoft Windows

Microsoft Windows is a group of several graphical operating system families, all of which are developed, marketed, and sold by Microsoft. Each family caters to a certain sector of the computing industry. Active Windows families include Windows NT and Windows Embedded; these may encompass subfamilies, e.g. Windows Embedded Compact (Windows CE) or Windows Server. Defunct Windows families include Windows 9x, Windows Mobile and Windows Phone.

Linux

Linux is a family of free and open-source software operating systems built around the Linux kernel. Typically, Linux is packaged in a form known as a Linux distribution (or *distro* for short) for both desktop and server use. The defining component of a Linux distribution is the Linux kernel, an operating system kernel first released on September 17, 1991, by Linus Torvalds.

Linux was originally developed for personal computers based on the Intel x86 architecture, but has since been ported to more platforms than any other operating system. Because of the dominance of the Linux kernel-based Android OS on smartphones, Linux has the largest installed base of all general-purpose operating systems.

UNIX SHELL

A **Unix shell** is a command-line interpreter or shell that provides a traditional Unix-like command line user interface. Users direct the operation of the computer by entering commands as text for a command line interpreter to execute, or by creating text scripts of one or more such commands. Users typically interact with a Unix shell using a terminal emulator, however, direct operation via serial hardware connections, or networking session, are common for server systems. All Unix shells provide filename wildcarding, piping, here documents, command substitution, variables and control structures for condition-testing and iteration.

BOURNE SHELL

The **Bourne shell** (**sh**) is a shell, or command-line interpreter, for computer operating systems.

The Bourne shell was the default shell for Version 7 Unix. Most Unix-like systems continue to have `/bin/sh`—which will be the Bourne shell, or a symbolic link or hard link to a compatible shell—even when other shells are used by most users.

Developed by Stephen Bourne at Bell Labs, it was a replacement for the Thompson shell, whose executable file had the same name—`sh`. It was released in 1979 in the Version 7 Unix release distributed to colleges and universities. Although it is used as an interactive command interpreter, it was also intended as a scripting language and contains most of the features that are commonly considered to produce structured programs.

KORN SHELL

Korn Shell (ksh) is a Unix shell which was developed by David Korn at Bell Labs in the early 1980s and announced at USENIX on July 14, 1983. The initial development was based on Bourne shell source code. Other early contributors were Bell Labs developers Mike Veach and Pat Sullivan, who wrote the Emacs and vi-style line editing modes' code, respectively. Korn Shell is backward-compatible with the Bourne shell and includes many features of the C shell, inspired by the requests of Bell Labs users.

C SHELL

The C shell, *csh*, was written by Bill Joy while a graduate student at University of California, Berkeley and widely distributed with BSD Unix. The language, including the control structures and the expression grammar, was modeled on C. The C shell also introduced a large number of features for interactive work, including the history and editing mechanisms, aliases, directory stacks, tilde notation, *cdpath*, job control and path hashing. On many systems, *csh* may be a symbolic link or hard link to TENEX C shell(*tcsh*), an improved version of Joy's original *csh*.

Though the C shell's interactive features have been copied in most other current shells, the language itself has not been widely copied. The only work-alike is Hamilton C shell, written by Nicole Hamilton, first distributed on OS/2 in 1988 and on Windows since 1992.

BASH SHELL

Bash is a Unix shell and command language written by Brian Fox for the GNU Project as a free software replacement for the Bourne shell. First released in 1989, it has been distributed widely as the default login shell for most Linux distributions and Apple's macOS (formerly OS X). A version is also available for Windows 10.

Bash is a command processor that typically runs in a text window, where the user types commands that cause actions. Bash can also read and execute commands from a file, called a shell script. Like all Unix shells, it supports filename globbing (wildcard matching), piping, here documents, command substitution, variables, and control structures for condition-testing and iteration. The keywords, syntax and other basic features of the language are all copied from *sh*. Other features, e.g., history, are copied from *csh* and *ksh*. Bash is a POSIX-compliant shell, but with a number of extensions.

The shell's name is an acronym for *Bourne-again shell*, a pun on the name of the Bourne shell that it replaces and on the term "born again" that denotes spiritual rebirth in Christian Evangelicalism.

Installing ubuntu 16.04

Firstly download the iso file from the internet of the ubuntu 16.04

Then make a bootable Pendrive or CD for installation

Then change the boot settings and boot from Pendrive or CD

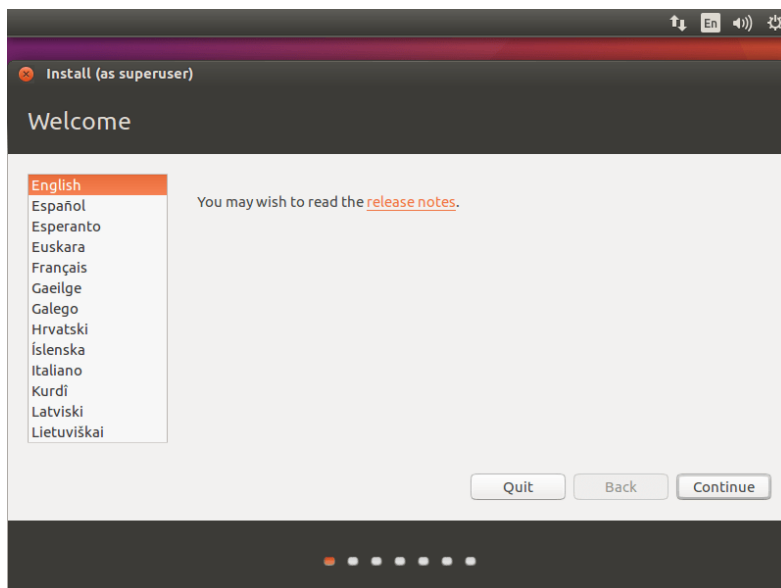
Now plug the installation medium and restart the system

After the restart the system will start booting from the installation medium

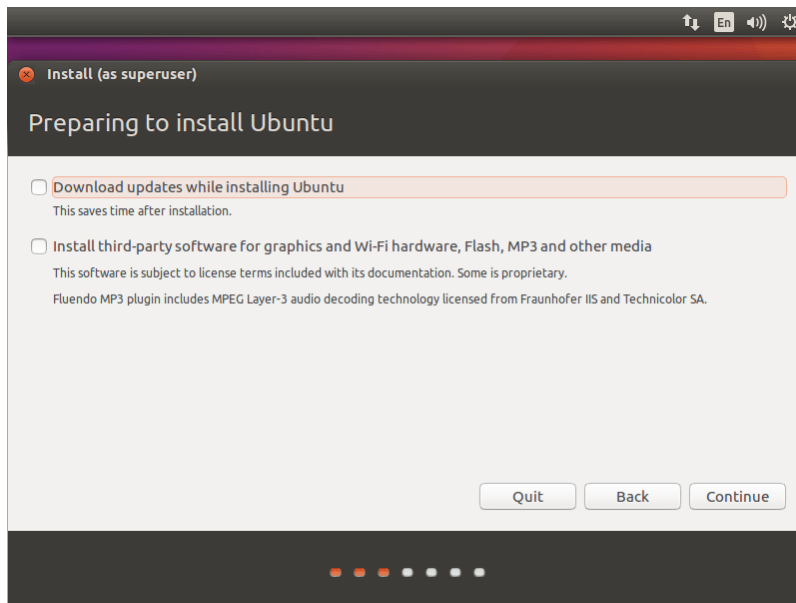


In the beginning of the installation it will ask for the language

Select the language and click the continue button



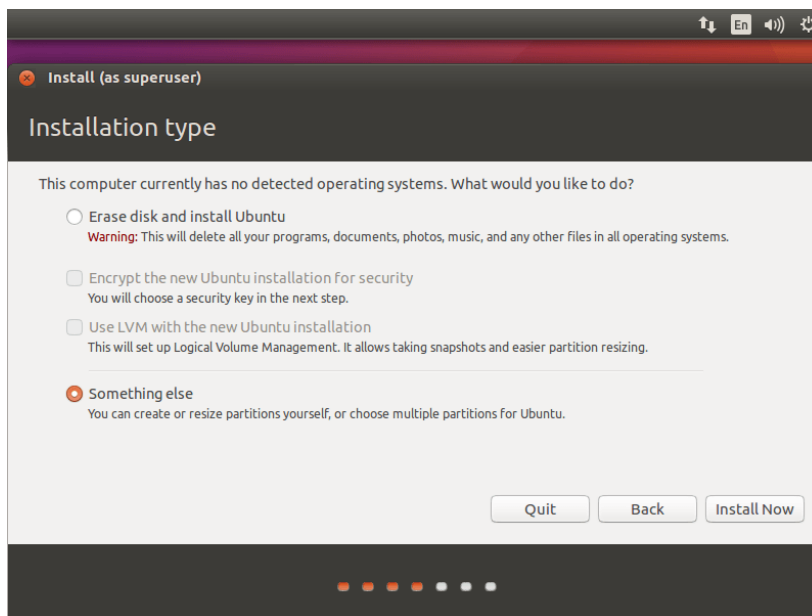
Then select the source of drivers while to install from internet alongside or to use the ones which come along the installation package



The next part is selecting the partition

Click on the something else option on the screen

Then click install now



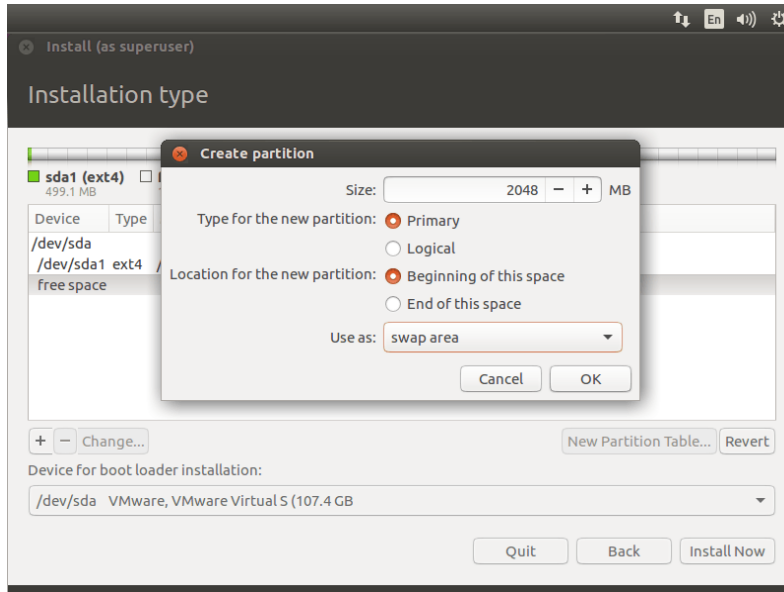
Now select the disk partition you intend to install Ubuntu

Be careful when undergoing these steps

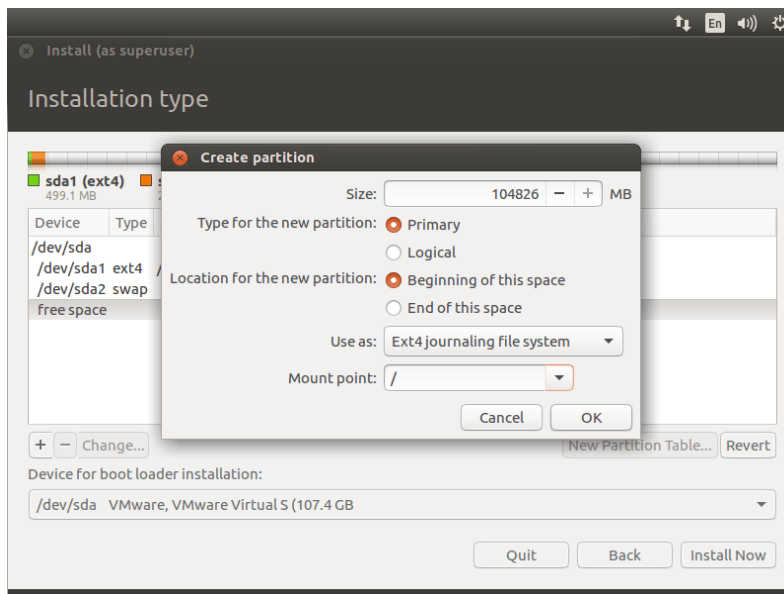
Now press the (-) button so as to delete that partition

Then (+) add that partition

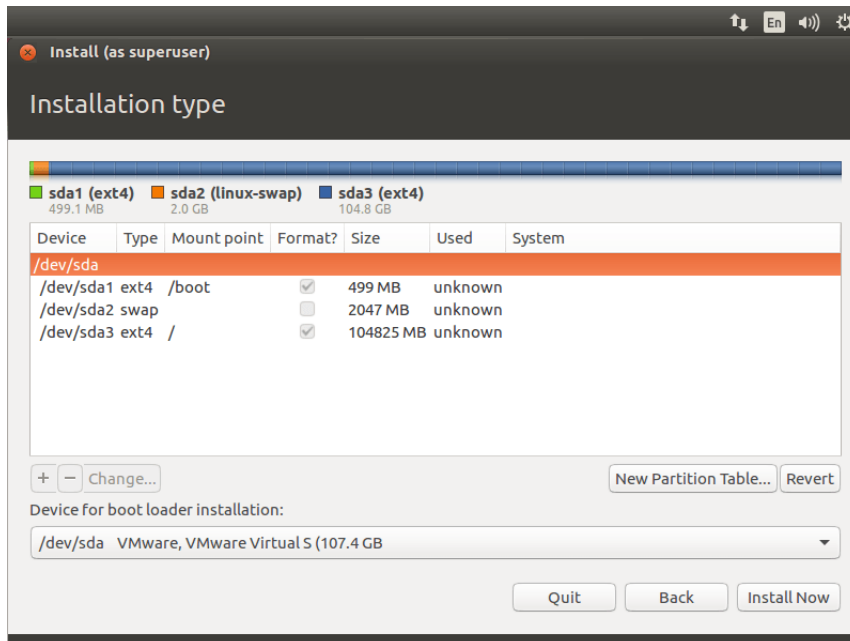
Then set size and for swap area and select the partition as swap area and press ok



Now select the rest of the partition for the ubuntu installation by selecting the mount point to be (/) then press ok

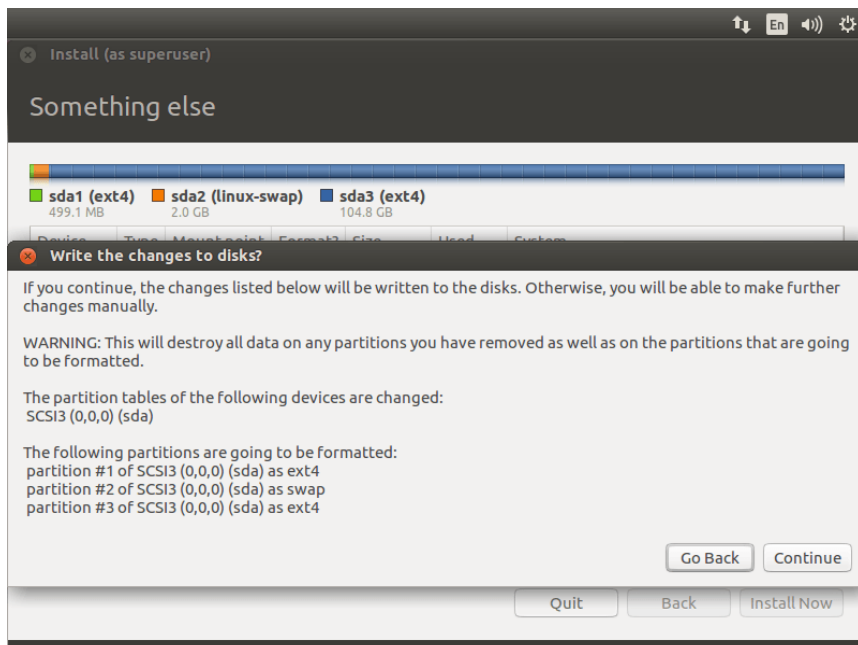


Review your partition layout and click on install now.



Write the changes to disk by clicking on continue.

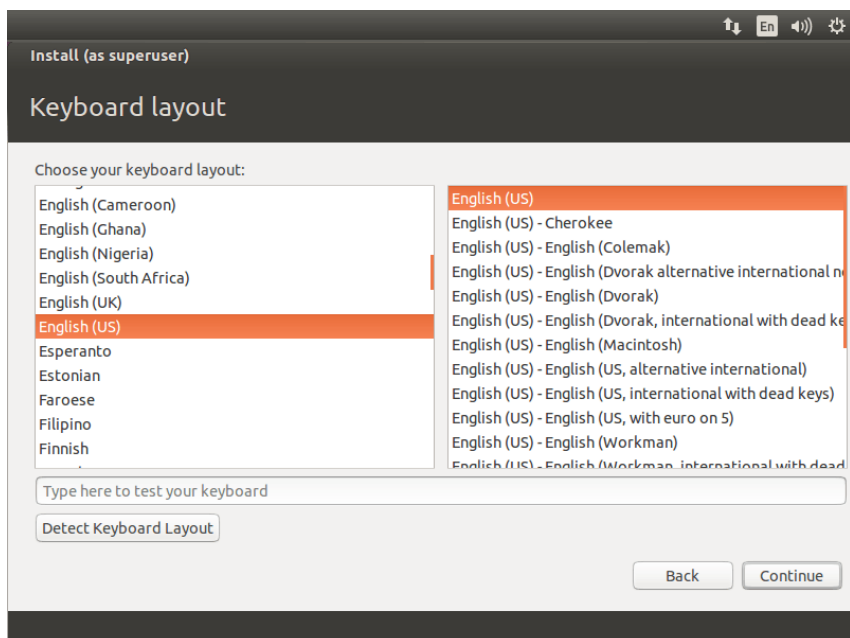
And take care as the changes can cause the loss of data in the drive



Select your location next screen.



Select your keyboard layout. If you are not sure, use the 'Detect Keyboard Layout' option. You can also test your selection by typing in the test text box.



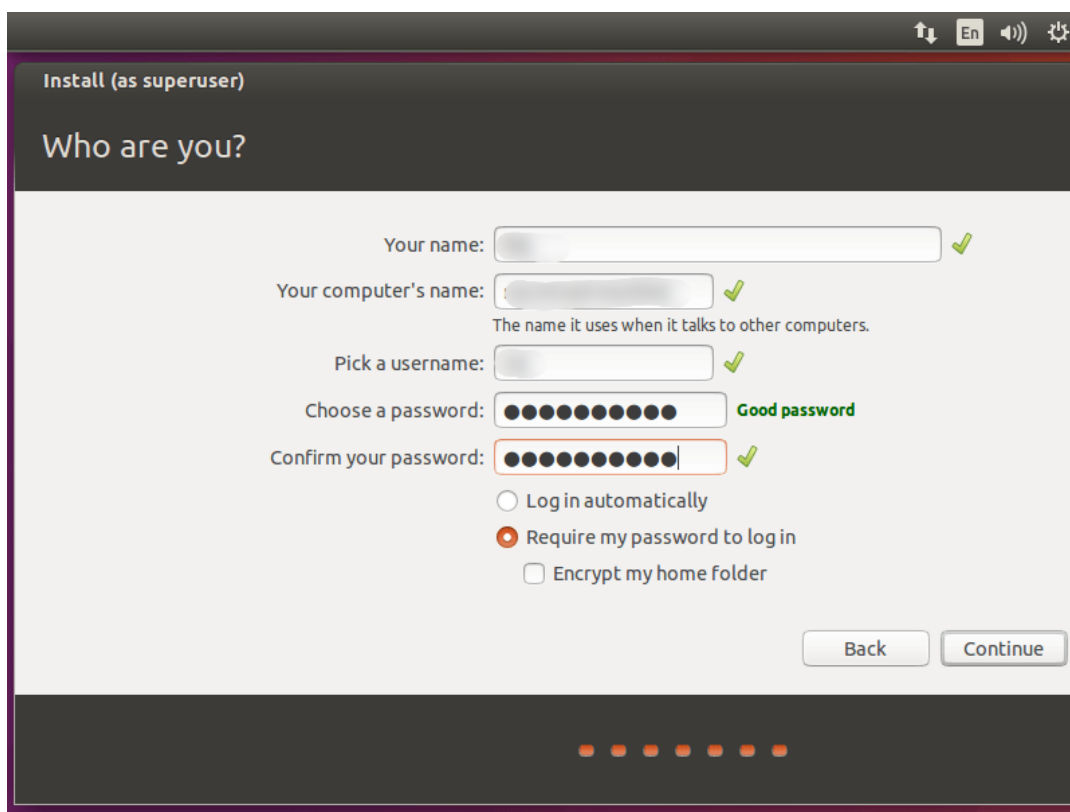
In the final screen of the installation wizard, you will be prompted to enter information about the user that you wanted to create on the system. Enter your information in this screen.

Here is one thing you should remember – if you select '**Login automatically**', System will directly take you to desktop without asking your credentials.

It's best if you give a very secure password for your installation. Ubuntu will tell you whether your password is secure or not.

If you select '**Encrypt my home folder**' it will make all the files and folders in your home folder more secure from unauthorized viewing if you have multiple users using your computer. When you log into your computer your files are seamlessly decrypted for just your session. If you are not sure, leave this box unchecked.

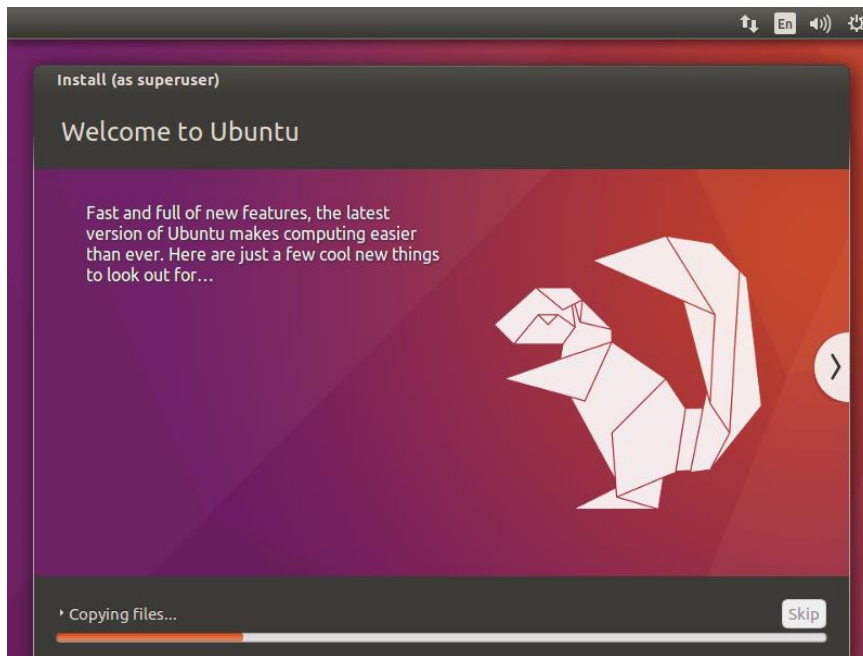
Once it's done, click on continue.



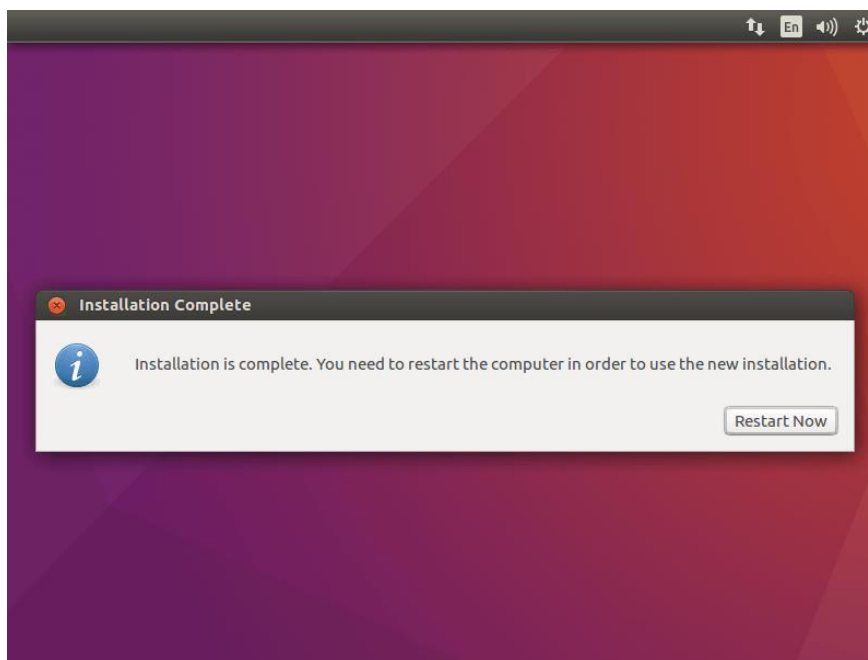
The screenshot shows the 'Install (as superuser)' window titled 'Who are you?'. It contains the following fields and options:

- Your name:** A text input field with a green checkmark to its right.
- Your computer's name:** A text input field with a green checkmark to its right. Below it, the text 'The name it uses when it talks to other computers.' is displayed.
- Pick a username:** A text input field with a green checkmark to its right.
- Choose a password:** A password input field (masked with dots) with the text 'Good password' in green to its right.
- Confirm your password:** A password input field (masked with dots) with a green checkmark to its right.
- Log in automatically:** An unselected radio button.
- Require my password to log in:** A selected radio button (indicated by a red dot).
- Encrypt my home folder:** An unselected checkbox.
- Buttons:** 'Back' and 'Continue' buttons are located at the bottom right.
- Progress indicator:** A row of seven dots is at the bottom, with the first five being orange and the last two being grey.

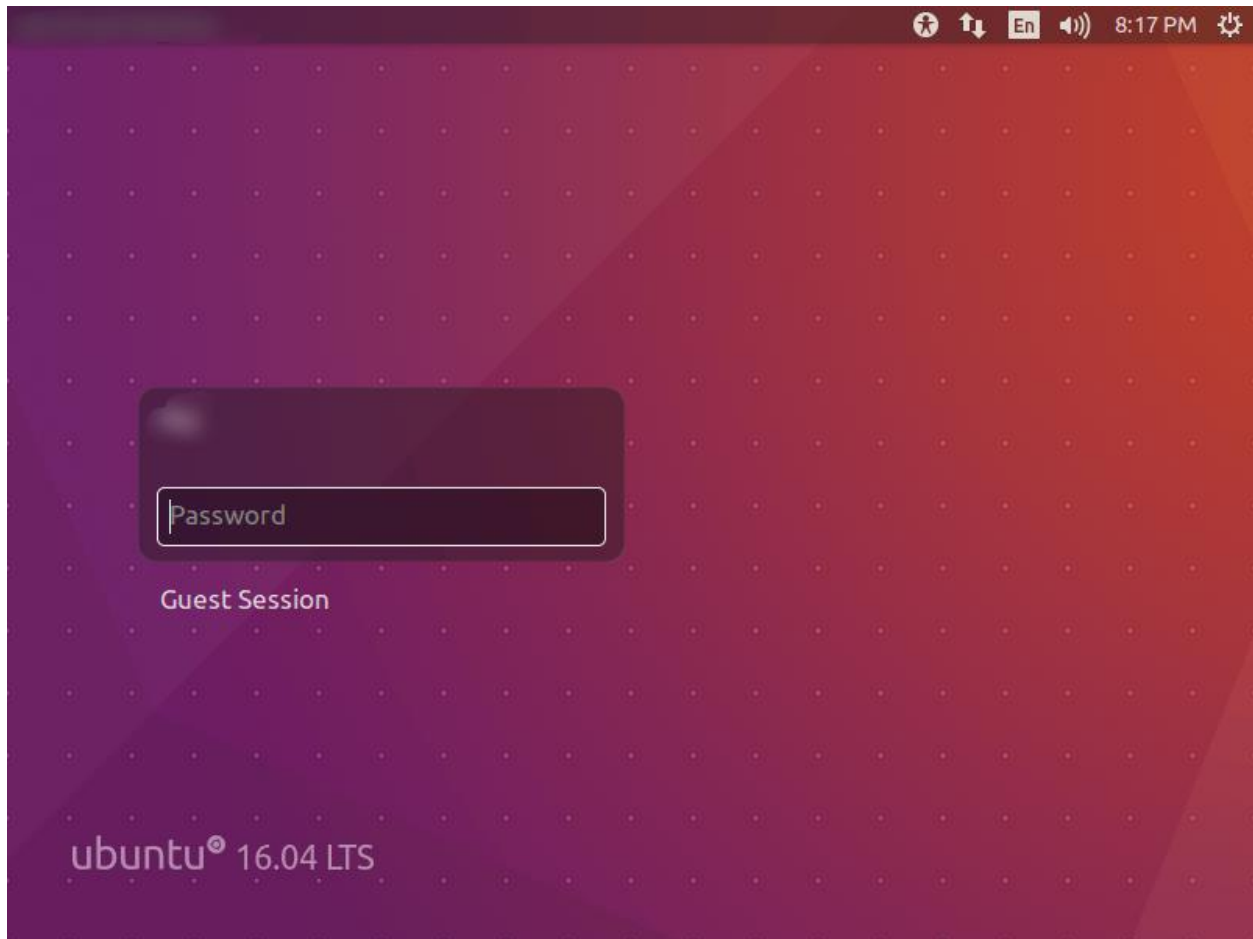
Below screenshot shows installing Ubuntu 16.04.



Once the installation is over, click on restart now.



Once your machine is restarted, you will get a login window. Login with username and password that you created earlier.



Practical 2

Commands : man, cat, touch, ls, script, cd, mkdir, rm, rmdir, pwd

man

Description : It displays manual page for the item(command)

Syntax \$ man [option(s)] keyword(s)

Options

-a	Display, in succession, all of the available intro manual pages contained within the manual. It is possible to quit between successive displays or skip any of them.
-C	Use this user configuration file
-k	Search the short descriptions and manual page names for the keyword as regular expression. Print out any matches.
-f	Lookup the manual pages referenced by name of command and print out the short descriptions of any found. Equivalent to whatis command
-w, --where, --path, -- location	print physical location of man page(s)
-K	-K, --global-apropos search for text in all pages

=> man command

shivank@shivank-Vostro-5568:~\$ man man

NAME

man - an interface to the on-line reference manuals

SYNOPSIS

```
man [-C file] [-d] [-D] [--warnings[=warnings]] [-R encoding] [-L locale] [-m system[,...]]
[-M path] [-S list] [-e extension] [-i|-I] [--regex|--wildcard] [--names-only] [-a] [-u]
[--no-subpages] [-P pager] [-r prompt] [-7] [-E encoding] [--no-hyphenation] [--no-justifi-
cation] [-p string] [-t] [-T[device]] [-H[browser]] [-X[dpi]] [-Z] [[section] page ...] ...
man -k [apropos options] regexp ...
```

:
:
:
:

HISTORY

1990, 1991 – Originally written by John W. Eaton (jwe@che.utexas.edu).

Dec 23 1992: Rik Faith (faith@cs.unc.edu) applied bug fixes supplied by Willem Kasdorp

(wkasdo@nikhef.knik.nl).

30th April 1994 – 23rd February 2000: Wilf. (G.Wilford@ee.surrey.ac.uk) has been developing and maintaining this package with the help of a few dedicated people.

30th October 1996 – 30th March 2001: Fabrizio Polacco <fpolacco@debian.org> maintained and enhanced this package for the Debian project, with the help of all the community.

31st March 2001 – present day: Colin Watson <cjwatson@debian.org> is now developing and maintaining man-db.

2.7.5

2015-11-06

MAN(1)

=>The output of man is shortened. The start and end of manual page of man command is shown above

1: -a

```
shivank@shivank-Vostro-5568:~$ man -a man
--Man-- next: man(7) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]
```

=> It showed two manual pages

2: -f

```
shivank@shivank-Vostro-5568:~$ man -f man
man (1)      - an interface to the on-line reference manuals
man (7)      - macros to format man pages
```

3: -w

```
shivank@shivank-Vostro-5568:~$ man -w cat
/usr/share/man/man1/cat.1.gz
shivank@shivank-Vostro-5568:~$ man --where cat
/usr/share/man/man1/cat.1.gz
shivank@shivank-Vostro-5568:~$ man --path cat
/usr/share/man/man1/cat.1.gz
shivank@shivank-Vostro-5568:~$ man --location cat
/usr/share/man/man1/cat.1.gz
```

4: -k

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ man -k cat
iconv_close (3)  - deallocate descriptor for character set conversion
iconv_open (3)  - allocate descriptor for character set conversion
```

```

xcb_alloc_color (3) - Allocate a color
xcb_alloc_color_reply (3) - Allocate a color
:
:
:
XtVaOpenApplication (3) - initialize, open, or close a display
XtWidgetToApplicationContext (3) - create, destroy, and obtain an application context
XVaCreateNestedList (3) - allocate a nested variable argument list
XWMHints (3) - allocate window manager hints structure and set or read a window's
WM_HINTS...
xzcat (1) - Compress or decompress .xz and .lzma files
zcat (1) - compress or expand files

```

=> In this way it shows all the things that contains the text 'cat'.

5: -K

```

shivank@shivank-Vostro-5568:~$ man -K cat
--Man-- next: odbcinst(1) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]

--Man-- next: python(1) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]

--Man-- next: python2(1) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]

--Man-- next: python2.7(1) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]

--Man-- next: xmlwf(1) [ view (return) | skip (Ctrl-D) | quit (Ctrl-C) ]
:
:
:
=> In this way it shows all the pages that contains the text.

```

cat

Description : cat is one of the most frequently used commands on Unix-like operating systems. It has three related functions with regard to text files: displaying them, combining copies of them and creating new ones.

Syntax `$ cat [options] [filenames] [-] [filenames]`

Options

-b	Displays the number along the lines in which something is written. No number shows when the line is empty
-e	It shows the end of line by replacing each end of line by \$

-n	It displays the number for each line outputted
-s	Suppress repeated empty output lines
-t	Displays TAB characters as ^I
-A	It is equivalent to the -e -t

=> cat command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat myfile
```

```
abc xyz pqr efg
```

```
xyz abc efg pqr
```

```
pqr efg abc xyz
```

```
hello world
```

```
in file 1
```

```
new line
```

=> It displays the content present in the file

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat > file1
```

=>After this command the content entered is the only content present in the file1

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2 > file1
```

=>This command copies the content of file2 and replaces the content of file1

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat >> file1
```

=>It appends the entered content to the file1

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2 >> file1
```

=>It appends the content of file 2 to file1

1: -b

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -b myfile
```

```
1 abc xyz pqr efg
```

```
2 xyz abc efg pqr
```

```
3 pqr efg abc xyz
```

```
4 hello world
```

```
5 in file 1
```

6 new line

2: -e

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -e myfile
```

```
abc xyz pqr efg$
```

```
xyz abc efg pqr$
```

```
pqr efg abc xyz$
```

```
$
```

```
$
```

```
$
```

```
hello world$
```

```
$
```

```
in file 1$
```

```
new line$
```

3: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -n myfile
```

```
1 abc xyz pqr efg
```

```
2 xyz abc efg pqr
```

```
3 pqr efg abc xyz
```

```
4
```

```
5
```

```
6
```

```
7 hello world
```

```
8
```

```
9 in file 1
```

```
10 new line
```

4: -s

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -s myfile
```

```
abc xyz pqr efg
```

```
xyz abc efg pqr
```

```
pqr efg abc xyz
```

```
hello world
```

```
in file 1
```

```
new line
```

5: -t

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -t myfile
```

```
abc xyz pqr efg
```

```
xyz abc efg pqr
```

```
pqr efg abc xyz
```

```
hello world
```

```
^lin^file^1
```

```
new line
```

6: -A

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat -A myfile
```

```
abc xyz pqr efg$
```

```
xyz abc efg pqr$
```

```
pqr efg abc xyz$
```

```
$
```

```
$
```

```
$
```

```
hello world$
```

```
$
```

```
^lin^file^1$
```

```
new line$
```

touch

Description : The *touch* command is the easiest way to create new, empty files. It is also used to change the *timestamps* (i.e., dates and times of the most recent access and modification) on existing files and directories.

Syntax \$ touch [option] file_name(s)

Options

-c	Do not create any files
-d	Parse STRING and use it instead of current time
-a	Change the access time only.
-m	Change the modification time only
-r	Use the times of the reference FILE instead of the current time.

=> touch command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch file1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Feb 12 19:46 file1
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

=>file created by the touch command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch file1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Feb 12 19:48 file1
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

=>Timestamp updated by the touch command

1: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch -c file2
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
```

```
file1  myfile
```

=>New file not created

2: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch -d '10-10-10' file1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Oct 10 2010 file1
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

=>Timestamp is set to the string

3: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lu
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Oct 10 2010 file1
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch -a file1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lu
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Feb 12 20:02 file1
```

```
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

=>Only access time of the file is modified

4: -m

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lc
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 0 Feb 12 19:52 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch -m file1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lc
total 4
-rw-rw-r-- 1 shivank shivank 0 Feb 12 20:01 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

=>The modification time of file is changed

5: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
total 4
-rw-rw-r-- 1 shivank shivank 0 Feb 12 20:01 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ touch -r myfile file1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
total 4
-rw-rw-r-- 1 shivank shivank 0 Feb 11 19:19 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
```

ls

Description : list directory contents

Syntax \$ ls [OPTION]... [FILE]...

Options

-a	--all : do not ignore entries starting with .
-A	--almost-all : do not list implied . and .. --author : with -l, print the author of each file
-b	--escape : print C-style escapes for nongraphic characters --block-size=SIZE : scale sizes by SIZE before printing them
-c	with -lt: sort by, and show, ctime (time of last modification of file status information) with -l: show ctime and sort by name; otherwise: sort by ctime, newest first
-C	list entries by columns --color[=WHEN] : colorize the output; WHEN can be 'always' (default if omitted), 'auto', or 'never'; more info below
-d	list directories themselves, not their contents
-D	generate output designed for Emacs' dired mode

-i	print the index number of each file
-m	fill width with a comma separated list of entries
-w	set output width to COLS. 0 means no limit

=>ls command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile practical1
```

1: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -a
. .. file 1 myfile practical1
```

2: -A

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -A
file 1 myfile practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lA
total 8
-rw-rw-r-- 1 shivank shivank 0 Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
```

3: -b

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -b
file\ 1 myfile practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l --block-size=KB
total 9kB
-rw-rw-r-- 1 shivank shivank 0kB Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank 1kB Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 1kB Feb 12 20:49 practical1
```

4: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l -c
total 8
-rw-rw-r-- 1 shivank shivank 0 Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lc
total 8
-rw-rw-r-- 1 shivank shivank 0 Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -ltc
```

```
total 8
-rw-rw-r-- 1 shivank shivank  0 Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
-rw-rw-r-- 1 shivank shivank  84 Feb 11 19:19 myfile
```

5: -C

```
shivank@shivank-Vostro-5568:~/Documents$ ls
a.sh imgpro NUP osfile pilog UE163095 WEb
shivank@shivank-Vostro-5568:~/Documents$ ls --color='auto'
a.sh imgpro NUP osfile pilog UE163095 WEb
shivank@shivank-Vostro-5568:~/Documents$ ls --color='always'
a.sh imgpro NUP osfile pilog UE163095 WEb
shivank@shivank-Vostro-5568:~/Documents$ ls --color='never'
a.sh imgpro NUP osfile pilog UE163095 WEb
```

6: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -ld
drwxrwxrwx 3 shivank shivank 4096 Feb 12 22:27 .
```

7: -D

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -lD
total 8
-rw-rw-r-- 1 shivank shivank  0 Feb 12 21:13 file 1
-rw-rw-r-- 1 shivank shivank  84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
//DIRED// 58 64 113 119 168 178
//DIRED-OPTIONS// --quoting-style=literal
```

8: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -i
6162339 file 1 6162349 myfile 6162340 practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l -i
total 8
6162339 -rw-rw-r-- 1 shivank shivank  0 Feb 12 21:13 file 1
6162349 -rw-rw-r-- 1 shivank shivank  84 Feb 11 19:19 myfile
6162340 -rw-rw-r-- 1 shivank shivank 417 Feb 12 20:49 practical1
```

9: -m

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -m
file 1, myfile, practical1
```

10: -w

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -w 30
file 1 myfile practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -w 10
```

file 1
myfile
practical1

script

Description : Make typescript of terminal session

Syntax \$ script [options] [file]

Options

-a	Append the output
-c	Run command rather than interactive shell
-q	Be quiet
-t	output timing data to stderr (or to FILE)
-f	Flush output after each write.

=>script command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ script practical1
Script started, file is practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file1 myfile practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ exit
Script done, file is practical1
```

1: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ script -a practical1
Script started, file is practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
total 8
-rw-rw-r-- 1 shivank shivank  0 Feb 11 19:19 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 421 Feb 12 20:24 practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ exit
exit
Script done, file is practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat practical1
Script started on Monday 12 February 2018 08:24:36 PM IST
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file1 myfile practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ exit
```

```
Script done on Monday 12 February 2018 08:24:56 PM IST
Script started on Monday 12 February 2018 08:25:12 PM IST
```



```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
total 8
-rw-rw-r-- 1 shivank shivank  0 Feb 11 19:19 file1
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
-rw-rw-r-- 1 shivank shivank 421 Feb 12 20:24 practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ exit
exit
```

Script done on Monday 12 February 2018 08:25:27 PM IST

2: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ script -c 'echo "Hello, World!"' file1
Script started, file is file1
Hello, World!
Script done, file is file1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file1
Script started on Monday 12 February 2018 08:40:36 PM IST
Hello, World!
```

Script done on Monday 12 February 2018 08:40:36 PM IST

3: -q

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ script -q -c 'echo "Hello, World!"' file1
Hello, World!
```

4: -t or --timing

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ script --timing='file1' practical1
Script started, file is practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file1  myfile  practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ exit
exit
Script done, file is practical1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file1
0.056002 130
5.496309 1
0.166058 1
0.368447 2
0.004838 27
0.002358 130
5.700491 1
```

cd**Description :** change directory**Syntax** \$ cd [option] [directory]**Options**

..	Goto parent directory
~	Goto user home
/	From root directory

=>cd command

shivank@shivank-Vostro-5568:~/Documents\$ cd UE163095/

shivank@shivank-Vostro-5568:~/Documents/UE163095\$

1: using /

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ cd /usr

shivank@shivank-Vostro-5568:/usr\$

2: using ..

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ cd ..

shivank@shivank-Vostro-5568:~/Documents\$

3: using ~

shivank@shivank-Vostro-5568:~/Documents\$ cd ~

shivank@shivank-Vostro-5568:~\$

mkdir**Description :** make directories**Syntax** \$ mkdir [OPTION]... DIRECTORY...**Options**

-m	set file mode (as in chmod), not a=rwx - umask
-v	print a message for each created directory
-p	no error if existing, make parent directories as needed

=>mkdir

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ mkdir ab

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ ls

ab myfile

1: -m

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ mkdir -m=rw sb

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls -l
total 8
-rw-rw-r-- 1 shivank shivank 84 Feb 11 19:19 myfile
drw-rw-r-- 2 shivank shivank 4096 Feb 13 13:06 sb
```

2: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mkdir -v sb
mkdir: created directory 'sb'
```

rm

Description : remove files or directories

Syntax \$ rm [OPTION]... [FILE]...

Options

-f	ignore nonexistent files and arguments, never prompt
-r	prompt before every removal
-i	remove directories and their contents recursively
-d	remove empty directories
-v	explain what is being done

=>rm

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
b myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm b
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

1: -f

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
b myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -f b
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
Myfile
```

2: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -r a
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

3: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -i -r a
rm: descend into directory 'a'? y
rm: descend into directory 'a/c'? y
rm: remove directory 'a/c/b'? y
rm: remove directory 'a/c'? y
rm: descend into directory 'a/b'? y
rm: remove directory 'a/b/c'? y
rm: remove directory 'a/b'? y
rm: remove directory 'a'? y
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

4: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -d a
rm: cannot remove 'a': Directory not empty
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -d a
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

5: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rm -r -v a
removed directory 'a/c/b'
removed directory 'a/c'
removed directory 'a/b/c'
removed directory 'a/b'
removed directory 'a'
```

rmdir

Description : Remove the DIRECTORY(ies), if they are empty.

Syntax \$ rmdir [OPTION]... DIRECTORY...

Options

-p	remove DIRECTORY and its ancestors; e.g., 'rmdir -p a/b/c' is similar to 'rmdir a/b/c a/b a'
-v	output a diagnostic for every directory processed

=>rmdir command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rmdir a
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

1: -p

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rmdir -p a/b/c
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
myfile
```

2: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ rmdir -p -v a/b/c
rmdir: removing directory, 'a/b/c'
rmdir: removing directory, 'a/b'
rmdir: removing directory, 'a'
```

pwd

Description : print name of current/working directory

Syntax \$ pwd [OPTION]...

Options

-L	The pathname printed will not contain symbolic links.
-P	The pathname printed can contain symbolic links

=>pwd

```
shivank@shivank-Vostro-5568:~$ pwd
/home/shivank
```

1: -L

```
shivank@shivank-Vostro-5568:~$ pwd -L
/home/shivank
```

2: -P

```
shivank@shivank-Vostro-5568:~$ pwd -P
/home/shivank
```

Practical 3

Commands : echo, who, whoami, cp, mv, tty, wc, tr, grep, tee

echo

Description : Display message on screen, writes each given STRING to standard output, with a space between each and a newline after the last one.

Syntax \$ echo [options]... [string]...

Options

-n	do not output the trailing newline
-E	disable interpretation of backslash escapes (default)
-e	enable interpretation of backslash escapes

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)

=>echo command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello \n world'
hello \n world
```

1: -E

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo -E 'hello \n world'
hello \n world
```

2: -e

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo -e 'hello \n world'
hello
world
```

3: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo -n 'hello \n world'
hello \n worldshivank@shivank-Vostro-5568:~/Documents/UE163095$
```

who

Description : Print information about users who are currently logged in.

Syntax \$ **who** [**OPTION**]... [**FILE** | **ARG1 ARG2**]

Options

-a	same as -b -d --login -p -r -t -T -u
-d	print dead processes
-H --heading	print line of column headings
-l	print system login processes
-q	all login names and number of users logged on
-r	print current runlevel
-s	print only name, line, and time (default)
-T	add user's message status as +, - or ?
-u	list users logged in

=> who

```
shivank@shivank-Vostro-5568:~$ who
shivank  tty7      2018-02-13 10:00 (:0)
```

1: -a

```
shivank@shivank-Vostro-5568:~$ who -a
      system boot  2018-02-13 10:00
      run-level 5  2018-02-13 10:00
LOGIN   tty1      2018-02-13 10:00      1133 id=tty1
shivank + tty7    2018-02-13 10:00 02:38    1343 (:0)
```

2: -d

```
shivank@shivank-Vostro-5568:~$ who -d
```

3: -H

```
shivank@shivank-Vostro-5568:~$ who -H
NAME    LINE    TIME      COMMENT
shivank tty7     2018-02-13 10:00 (:0)
```

4: -l

```
shivank@shivank-Vostro-5568:~$ who -l
LOGIN   tty1      2018-02-13 10:00      1133 id=tty1
```

5: -q

```
shivank@shivank-Vostro-5568:~$ who -q
shivank
# users=1
```

6: -r

```
shivank@shivank-Vostro-5568:~$ who -r
      run-level 5  2018-02-13 10:00
```

7: -s

```
shivank@shivank-Vostro-5568:~$ who -s
shivank tty7     2018-02-13 10:00 (:0)
```

8: -T

```
shivank@shivank-Vostro-5568:~$ who -T
shivank + tty7    2018-02-13 10:00 (:0)
```

9: -u

```
shivank@shivank-Vostro-5568:~$ who -u
shivank tty7     2018-02-13 10:00 02:44    1343 (:0)
```


whoami**Description :** Print the current user id and name.**Syntax** \$ **whoami** [**options**]**Options**

--help	display this help and exit
--version	output version information and exit

=>whoami command

```
shivank@shivank-Vostro-5568:~$ whoami
shivank
```

1: --help

```
shivank@shivank-Vostro-5568:~$ whoami --help
```

Usage: whoami [OPTION]...

Print the user name associated with the current effective user ID.

Same as id -un.

```
--help    display this help and exit
--version  output version information and exit
```

GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>Full documentation at: <<http://www.gnu.org/software/coreutils/whoami>>

or available locally via: info '(coreutils) whoami invocation'

2: --version

```
shivank@shivank-Vostro-5568:~$ whoami --version
```

whoami (GNU coreutils) 8.25

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 Richard Mlynarik.

cp**Description :** Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.**Syntax** \$ **cp** [**OPTION**]... [**-T**] **SOURCE DEST**or: **cp** [**OPTION**]... **SOURCE... DIRECTORY**or: **cp** [**OPTION**]... **-t DIRECTORY SOURCE...****Options**

-R, -r, --recursive	copy directories recursively
----------------------------	------------------------------

-i, --interactive	prompt before overwrite (overrides a previous -n option)
-v, --verbose	explain what is being done
-n, --no-clobber	do not overwrite an existing file (overrides a previous -i option)
-u, --update	copy only when the SOURCE file is newer than the destination file or when the destination file is missing

=>cp command

```

shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp file file2
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file file2 myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file
34
65
23
12
76
32
98
97
355
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
34
65
23
12
76
32
98
97
355

```

=>All the contents of the file is copied to the file2 that is created in the same directory

1: -r

```

shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp a m
cp: omitting directory 'a'
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file file2 myfile myfile1

```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp -r a m
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file file2 m myfile myfile1
```

=> The directories can't be copied directly thus they are copied recursively by using option -r

2: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file file2 myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp -i file file2
cp: overwrite 'file2'? y
```

3: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp -i -v file file2
cp: overwrite 'file2'? y
'file' -> 'file2'
shivank@shivank-Vo
```

4: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file file2 myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat myfile1
hello world    in file 2
end of the file myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file
34
355
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cp -n file myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat myfile1
hello world    in file 2
end of the file myfile1
```

mv

Description : Rename SOURCE to DEST, or move SOURCE(s) to DIRECTORY.

Syntax \$ mv [OPTION]... [-T] SOURCE DEST

or: mv [OPTION]... SOURCE... DIRECTORY

or: mv [OPTION]... -t DIRECTORY SOURCE...

Options

-f, --force	do not prompt before overwriting
-i, --interactive	prompt before overwrite
-n, --no-clobber	do not overwrite an existing file

-u, --update	move only when the SOURCE file is newer than the destination file or when the destination file is missing
-v, --verbose	explain what is being done
-T, --no-target-directory	treat DEST as a normal file

=> mv command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
b file file2 myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mv b a
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file file2 myfile myfile1
```

1: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
50
40
30
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mv -i file file2
mv: overwrite 'file2'? y
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
34
355
```

=>It prompts before overwriting a file

2: -f

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
50
40
30
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mv -f file file2
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
34
355
```

=>This option does not prompts before overwriting

3: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a file2 myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mv -v a m
```

'a' -> 'm'

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file2 m myfile myfile1
```

4: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
file2 m myfile myfile1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat myfile
abc xyz pqr efg
xyz abc efg pqr
pqr efg abc xyz
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat file2
50
40
30
shivank@shivank-Vostro-5568:~/Documents/UE163095$ mv -n file2 myfile
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat myfile
abc xyz pqr efg
xyz abc efg pqr
pqr efg abc xyz
```

tty

Description : Print the file name of the terminal connected to standard input.

Syntax \$ **tty** [**options**]

Options

-s --silent --quiet	print nothing, only return an exit status
--version	output version information and exit
--help	display this help and exit

=>tty command

```
shivank@shivank-Vostro-5568:~$ tty
/dev/pts/21
```

1: -s

```
shivank@shivank-Vostro-5568:~$ tty -s
shivank@shivank-Vostro-5568:~$ tty --silent
shivank@shivank-Vostro-5568:~$ tty --quiet
```

2: --version

```
shivank@shivank-Vostro-5568:~$ tty --version
```

```
tty (GNU coreutils) 8.25
```

```
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 David MacKenzie.

3: --help

```
shivank@shivank-Vostro-5568:~$ tty --help
```

```
Usage: tty [OPTION]...
```

```
Print the file name of the terminal connected to standard input.
```

```
-s, --silent, --quiet  print nothing, only return an exit status
```

```
--help  display this help and exit
```

```
--version  output version information and exit
```

GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>

Full documentation at: <<http://www.gnu.org/software/coreutils/tty>>

or available locally via: info '(coreutils) tty invocation'

wc

Description : Print byte, word, and line counts, count the number of bytes, whitespace-separated words, and newlines in each given FILE, or standard input if none are given or for a FILE of `-'.

Syntax \$ wc [options]... [file]...

Options

-c, --bytes	print the byte counts
-m, --chars	print the character counts
-l, --lines	print the newline counts
-L, --max-line-length	print the maximum display width
-w	print the word counts

=> wc command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc myfile
```

```
10 19 84 myfile
```

1: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc -c myfile
```

```
84 myfile
```

2: -m

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc -m myfile
84 myfile
```

3: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc -l myfile
10 myfile
```

4: -L

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc -L myfile
25 myfile
```

5: -w

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ wc -w myfile
19 myfile
```

tr

Description : Translate, squeeze, and/or delete characters from standard input, writing to standard output.

Syntax \$ tr [OPTION]... SET1 [SET2]

Options

-c, -C, --complement	use the complement of SET1
-d, --delete	delete characters in SET1, do not translate
-s, --squeeze-repeats	replace each sequence of a repeated character that is listed in the last specified SET, with a single occurrence of that character
-t, --truncate-set1	first truncate SET1 to length of SET2

=>tr command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr a-z A-Z
HELLO
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr 'ello' 'pqr'
hprrr
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr 'elo' 'pqrs'
hpqqr
```

=>It replaces all the letters with the corresponding letters in the set2

1: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr -c 'elo' 'pqrs'
sellos
```

2: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr -d 'el'
ho
```

3: -s

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr 'elo' 'pqrs'
hpqqr
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr -s 'elo' 'pqrs'
hpqr
```

4: -t

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr 'elo' 'pq'
hpqqq
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello' | tr -t 'elo' 'pq'
hpqqo
```

grep

Description : print lines matching a pattern

Syntax \$ **grep** [OPTION]... **PATTERN** [FILE]...

Options

-A NUM, --after-context=NUM	Print NUM lines of trailing context after matching lines.
-a, --text	Process a binary file as if it were text
-B NUM, --before-context=NUM	Print NUM lines of leading context before matching lines.
-C, --context=NUM	print NUM lines of output context
-c, --count	print only a count of matching lines per FILE

=> grep command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep microcomputer
/usr/share/dict/words
microcomputer
microcomputer's
microcomputers
```

1: -A

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep -A 2 microcomputer
/usr/share/dict/words
microcomputer
microcomputer's
```


microcomputers
microcosm
microcosm's

2: -B

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep -B 2 microcomputer  
/usr/share/dict/words  
microchips  
microcode  
microcomputer  
microcomputer's  
microcomputers
```

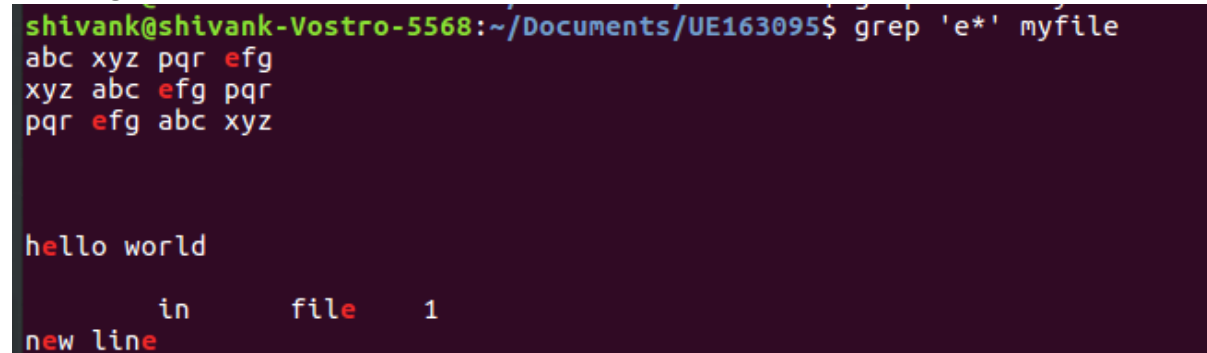
3: -C

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep -C 2 microcomputer  
/usr/share/dict/words  
microchips  
microcode  
microcomputer  
microcomputer's  
microcomputers  
microcosm  
microcosm's
```

4: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep -c microcomputer  
/usr/share/dict/words  
3
```

5: using '*'



```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ grep 'e*' myfile  
abc xyz pqr efg  
xyz abc efg pqr  
pqr efg abc xyz  
  
hello world  
  
in file 1  
new line
```

tee

Description : Copy standard input to each FILE, and also to standard output.

Syntax \$ tee [OPTION]... [FILE]...

Options

-a, --append	append to the given FILEs, do not overwrite
-i, --ignore-interrupts	ignore interrupt signals
-p	diagnose errors writing to non pipes
--output-error[=MODE]	set behavior on write error. See MODE below

=>tee command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
```

```
file2 m myfile myfile1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls | wc -l | tee count.txt
```

```
5
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
```

```
count.txt file2 m myfile myfile1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat count.txt
```

```
5
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls | wc | tee -a count.txt
```

```
5    5   33
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cat count.txt
```

```
5
```

```
5    5   33
```

Practical 4

Commands : cmp, comm, diff, df, du, free, whatis, whereis, find, type

cmp

Description : compare two bytes byte by byte

Syntax \$ **cmp** [**OPTION**]... **FILE1** [**FILE2** [**SKIP1** [**SKIP2**]]]

Options

-b	print differing bytes
-i	--ignore-initial=SKIP : skip first SKIP bytes of both inputs --ignore-initial=SKIP1:SKIP2 : skip first SKIP1 bytes of FILE1 and first SKIP2 bytes of FILE2
-l	--verbose : output byte numbers and differing byte values
-s	--quiet, --silent : suppress all normal output
-n	--bytes=LIMIT : compare at most LIMIT bytes

=>cmp command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp myfile myfile1
myfile myfile1 differ: byte 1, line 1
```

1: -b

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp -b myfile myfile1
myfile myfile1 differ: byte 1, line 1 is 141 a 150 h
```

2: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp --ignore-initial=3 myfile myfile1
myfile myfile1 differ: byte 1, line 1
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp --ignore-initial=2:3 myfile myfile1
myfile myfile1 differ: byte 1, line 1
```

3: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp -l myfile myfile1
1 141 150
2 142 145
3 143 154
:
:
```

4: -s

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp -s myfile myfile1
```

5: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cmp --bytes=4 myfile myfile1
myfile myfile1 differ: byte 1, line 1
```

comm

Description : Compare sorted files FILE1 and FILE2 line by line.

Syntax \$ comm [OPTION]... FILE1 FILE2

Options

-1	suppress column 1 (lines unique to FILE1)
-2	suppress column 2 (lines unique to FILE2)
-3	suppress column 3 (lines that appear in both files)
--check-order	check that the input is correctly sorted, even if all input lines are pairable
--nocheck-order	do not check that the input is correctly sorted
-z,--zero-terminated	line delimiter is NUL, not newline

=>comm command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm myfile myfile1
```

```
abc xyz pqr efg
```

```
hello world in file 2
```

```
comm: file 2 is not in sorted order
```

```
abc xyz
pqr efg
new file
```

```
end of the file myfile1
```

```
xyz abc efg pqr
```

```
comm: file 1 is not in sorted order
```

```
pqr efg abc xyz
```

```
hello world
```

```
in file 1
```

```
new line
```

1: Options 1,2 and 3

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm -1 myfile myfile1
```

```
hello world    in file 2
```

```
comm: file 2 is not in sorted order
```

```
abc           xyz
```

```
    pqr       efg
```

```
new file
```

```
end of the file myfile1
```

```
comm: file 1 is not in sorted order
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm -2 myfile myfile1
```

```
abc xyz pqr efg
```

```
comm: file 2 is not in sorted order
```

```
xyz abc efg pqr
```

```
comm: file 1 is not in sorted order
```

```
pqr efg abc xyz
```

```
hello world
```

```
    in    file    1
```

```
new line
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm -3 myfile myfile1
```

```
abc xyz pqr efg
```

```
    hello world    in file 2
```

```
comm: file 2 is not in sorted order
```

```
abc           xyz
```

```
    pqr       efg
```

```
new file
```

```
end of the file myfile1
```

```
xyz abc efg pqr
```

```
comm: file 1 is not in sorted order
```

```
pqr efg abc xyz
```

```
hello world
```

```

        in      file      1
new line

```

2: --nocheck-order

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm --nocheck-order myfile myfile1
```

```

abc xyz pqr efg
      hello world    in file 2

      abc            xyz
        pqr          efg
new file

```

```

        end of the file myfile1
xyz abc efg pqr
pqr efg abc xyz

```

```
hello world
```

```

        in      file      1
new line

```

3: --check-order

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm --check-order myfile myfile1
```

```

abc xyz pqr efg
      hello world    in file 2
comm: file 2 is not in sorted order

```

4: -z

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ comm -z myfile myfile1
```

```

abc xyz pqr efg
xyz abc efg pqr
pqr efg abc xyz

```

```
hello world
```

```

        in      file      1
new line
      hello world    in file 2

```

```

abc      xyz
      pqr      efg
new file

```

```

end of the file myfile1
s

```

diff

Description : Compare FILES line by line.

Syntax \$ diff [OPTION]... FILES

Options

-a, --text	Treat all files as text.
-b, --ignore-space-change	Ignore changes in the amount of white space.
-B, --ignore-blank-lines	Ignore changes whose lines are all blank.
-q, --brief	report only when files differ
-d, --minimal	Try hard to find a smaller set of changes.

=> diff command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ diff myfile myfile1
```

```

1,3c1
< abc xyz pqr efg
< xyz abc efg pqr
< pqr efg abc xyz
---
> hello world  in file 2
4a3,5
> abc      xyz
>      pqr      efg
> new file
7,10c8
< hello world
<
<      in      file      1
< new line
---
> end of the file myfile1

```

1: -q

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ diff -q myfile myfile1
Files myfile and myfile1 differ

2: -d

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ diff -d myfile myfile1
1,4c1
< abc xyz pqr efg
< xyz abc efg pqr
< pqr efg abc xyz
<

> hello world in file 2
5a3,5
> abc xyz
> pqr efg
> new file
7d6
< hello world
9,10c8
< in file 1
< new line

> end of the file myfile1

3: -b

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ diff -b myfile myfile1
1,3c1
< abc xyz pqr efg
< xyz abc efg pqr
< pqr efg abc xyz

> hello world in file 2
4a3,5
> abc xyz
> pqr efg
> new file
7,10c8
< hello world
< in file 1
< new line

> end of the file myfile1

df

Description : Show information about the file system on which each FILE resides, or all file systems by default.

Syntax \$ df [OPTION]... [FILE]...

Options

-a	include pseudo, duplicate, inaccessible file systems
-B	--block-size=SIZE scale sizes by SIZE before printing them; e.g., '-BM' prints sizes in units of 1,048,576 bytes;
-h	--human-readable print sizes in powers of 1024 (e.g., 1023M)
-H	print sizes in powers of 1000 (e.g., 1.1G)
-i	--inodes list inode information instead of block usage
-P	use the POSIX output format
-t	limit listing to file systems of type TYPE

=>df command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ df
Filesystem    1K-blocks    Used Available Use% Mounted on
udev          4002376      0 4002376  0% /dev
tmpfs         806472    9764 796708  2% /run
/dev/sda7     98251396 21311004 71926428 23% /
tmpfs         4032340  509036 3523304 13% /dev/shm
tmpfs         5120        4  5116  1% /run/lock
tmpfs         4032340      0 4032340  0% /sys/fs/cgroup
/dev/sda1      507904    73740 434164 15% /boot/efi
tmpfs         806472      72 806400  1% /run/user/1000
/dev/sda5     422010876 160150124 261860752 38% /media/shivank/SB
```

1: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ df -a
Filesystem    1K-blocks    Used Available Use% Mounted on
sysfs         0          0      0 - /sys
proc          0          0      0 - /proc
udev          4002376      0 4002376  0% /dev
devpts        0          0      0 - /dev/pts
tmpfs         806472    9764 796708  2% /run
/dev/sda7     98251396 21311004 71926428 23% /
securityfs    0          0      0 - /sys/kernel/security
```

```

tmpfs      4032340 509036 3523304 13% /dev/shm
tmpfs      5120    4    5116 1% /run/lock
tmpfs      4032340    0 4032340 0% /sys/fs/cgroup
cgroup      0    0    0 - /sys/fs/cgroup/systemd
pstore      0    0    0 - /sys/fs/pstore
efivarfs    0    0    0 - /sys/firmware/efi/efivars
cgroup      0    0    0 - /sys/fs/cgroup/freezer
cgroup      0    0    0 - /sys/fs/cgroup/cpu,cpuacct
cgroup      0    0    0 - /sys/fs/cgroup/hugetlb
cgroup      0    0    0 - /sys/fs/cgroup/net_cls,net_prio
cgroup      0    0    0 - /sys/fs/cgroup/blkio
cgroup      0    0    0 - /sys/fs/cgroup/pids
cgroup      0    0    0 - /sys/fs/cgroup/memory
cgroup      0    0    0 - /sys/fs/cgroup/perf_event
cgroup      0    0    0 - /sys/fs/cgroup/cpuset
cgroup      0    0    0 - /sys/fs/cgroup/rdma
cgroup      0    0    0 - /sys/fs/cgroup/devices
systemd-1    0    0    0 - /proc/sys/fs/binfmt_misc
hugetlbfs    0    0    0 - /dev/hugepages
mqueue      0    0    0 - /dev/mqueue
debugfs      0    0    0 - /sys/kernel/debug
fusectl      0    0    0 - /sys/fs/fuse/connections
configfs     0    0    0 - /sys/kernel/config
/dev/sda1    507904 73740 434164 15% /boot/efi
tmpfs      806472    72 806400 1% /run/user/1000
gvfsd-fuse    0    0    0 - /run/user/1000/gvfs
/dev/sda5   422010876 160150124 261860752 38% /media/shivank/SB

```

2: -B

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ df -BG

```
Filesystem 1G-blocks Used Available Use% Mounted on
```

```

udev      4G 0G 4G 0% /dev
tmpfs     1G 1G 1G 2% /run
/dev/sda7 94G 21G 69G 23% /
tmpfs     4G 1G 4G 13% /dev/shm
tmpfs     1G 1G 1G 1% /run/lock
tmpfs     4G 0G 4G 0% /sys/fs/cgroup
/dev/sda1 1G 1G 1G 15% /boot/efi
tmpfs     1G 1G 1G 1% /run/user/1000
/dev/sda5 403G 153G 250G 38% /media/shivank/SB

```

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ df -BM

```
Filesystem 1M-blocks Used Available Use% Mounted on
```

```

udev          3909M    0M   3909M  0% /dev
tmpfs         788M    10M   779M  2% /run
/dev/sda7     95949M 20812M 70241M 23% /
tmpfs         3938M   498M   3441M 13% /dev/shm
tmpfs         5M      1M     5M   1% /run/lock
tmpfs         3938M    0M   3938M  0% /sys/fs/cgroup
/dev/sda1     496M    73M   424M 15% /boot/efi
tmpfs         788M    1M     788M  1% /run/user/1000
/dev/sda5     412120M 156397M 255724M 38% /media/shivank/SB

```

3: -h

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
udev	3.9G	0	3.9G	0%	/dev
tmpfs	788M	9.6M	779M	2%	/run
/dev/sda7	94G	21G	69G	23%	/
tmpfs	3.9G	497M	3.4G	13%	/dev/shm
tmpfs	5.0M	4.0K	5.0M	1%	/run/lock
tmpfs	3.9G	0	3.9G	0%	/sys/fs/cgroup
/dev/sda1	496M	73M	424M	15%	/boot/efi
tmpfs	788M	72K	788M	1%	/run/user/1000
/dev/sda5	403G	153G	250G	38%	/media/shivank/SB

4: -i

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ df -i
```

Filesystem	Inodes	IUsed	IFree	IUse%	Mounted on
udev	1000594	593	1000001	1%	/dev
tmpfs	1008085	909	1007176	1%	/run
/dev/sda7	6250496	428402	5822094	7%	/
tmpfs	1008085	183	1007902	1%	/dev/shm
tmpfs	1008085	5	1008080	1%	/run/lock
tmpfs	1008085	17	1008068	1%	/sys/fs/cgroup
/dev/sda1	0	0	0	-	/boot/efi
tmpfs	1008085	33	1008052	1%	/run/user/1000
/dev/sda5	261959056	72720	261886336	1%	/media/shivank/SB

5: -P

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ df -P
```

Filesystem	1024-blocks	Used	Available	Capacity	Mounted on
udev	4002376	0	4002376	0%	/dev
tmpfs	806472	9764	796708	2%	/run
/dev/sda7	98251396	21311252	71926180	23%	/
tmpfs	4032340	508880	3523460	13%	/dev/shm
tmpfs	5120	4	5116	1%	/run/lock

```
tmpfs      4032340      0 4032340      0% /sys/fs/cgroup
/dev/sda1   507904    73740 434164      15% /boot/efi
tmpfs      806472      72 806400       1% /run/user/1000
/dev/sda5  422010876 160150124 261860752    38% /media/shivank/SB
```

du

Description : Summarize disk usage of the set of FILEs, recursively for directories.

Syntax \$ du [OPTION]... [FILE]...

Options

-a	write counts for all files, not just directories
-B	--block-size=SIZE scale sizes by SIZE before printing them; e.g., '-BM' prints sizes in units of 1,048,576 bytes;
-b	equivalent to '--apparent-size --block-size=1'
-c	produce a grand total
-D	dereference only symlinks that are listed on the command line
-h	print sizes in human readable format (e.g., 1K 234M 2G)
-H	equivalent to --dereference-args (-D)

=>du command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du
```

```
4      ./a/c
4      ./a/b/c
8      ./a/b
16     ./a
4      ./b
28     .
```

1: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du -a
```

```
4      ./a/c
4      ./a/b/c
8      ./a/b
16     ./a
4      ./b
4      ./myfile
28     .
```

2: -b

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du -b
```

```
4096  ./a/c
4096  ./a/b/c
8192  ./a/b
16384 ./a
4096  ./b
24660 .
```

3: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du -c
```

```
4      ./a/c
4      ./a/b/c
8      ./a/b
16     ./a
4      ./b
28     .
28     total
```

4: -D

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du -D
```

```
4      ./a/c
4      ./a/b/c
8      ./a/b
16     ./a
4      ./b
28     .
```

5: -h

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ du -h
```

```
4.0K   ./a/c
4.0K   ./a/b/c
8.0K   ./a/b
16K    ./a
4.0K   ./b
28K    .
```

free**Description :** Display amount of free and used memory in the system**Syntax** \$ free [options]**Options**

-b, --bytes	Show output in bytes
-k, --kilo	Show output in kilobytes
-m, --mega	Show output in megabytes
-g, --giga --tera	Show output in gigabytes Show output in terabytes
-h, --human	Show human-readable output
-l, --lohi	Show detailed low and high memory statistics
-t, --total	Show total for RAM + swap
-c N, --count N	Repeat printing N times, then exit
-s N, --seconds N	Repeat printing every N seconds
-w, --wide	Wide output

=>free command

shivank@shivank-Vostro-5568:~\$ free

```

      total    used    free   shared  buff/cache   available
Mem:   8064676 1391328  5103564   431516   1569784   6148628
Swap:   4882428      0  4882428

```

1: -b

shivank@shivank-Vostro-5568:~\$ free -b

```

      total    used    free   shared  buff/cache   available
Mem:  8258228224 1388302336 5261926400  438013952 1607999488 6323916800
Swap: 4999606272      0 4999606272

```

2: -k

shivank@shivank-Vostro-5568:~\$ free -k

```

      total    used    free   shared  buff/cache   available
Mem:   8064676 1354468  5139872   427748   1570336   6177136

```

3: -m

shivank@shivank-Vostro-5568:~\$ free -m

	total	used	free	shared	buff/cache	available
Mem:	7875	1322	5019	417	1533	6032
Swap:	4767	0	4767			

4: -g

shivank@shivank-Vostro-5568:~\$ free -g

	total	used	free	shared	buff/cache	available
Mem:	7	1	4	0	1	5
Swap:	4	0	4			

5: --tera

shivank@shivank-Vostro-5568:~\$ free --tera

	total	used	free	shared	buff/cache	available
Mem:	0	0	0	0	0	0
Swap:	0	0	0			

6: -h

shivank@shivank-Vostro-5568:~\$ free -h

	total	used	free	shared	buff/cache	available
Mem:	7.7G	1.3G	4.9G	416M	1.5G	5.9G
Swap:	4.7G	0B	4.7G			

7: -l

shivank@shivank-Vostro-5568:~\$ free -l

	total	used	free	shared	buff/cache	available
Mem:	8064676	1343456	5152476	426112	1568744	6189892
Low:	8064676	2912200	5152476			
High:	0	0	0			
Swap:	4882428	0	4882428			

8: -t

shivank@shivank-Vostro-5568:~\$ free -t

	total	used	free	shared	buff/cache	available
Mem:	8064676	1343608	5152220	426192	1568848	6189656
Swap:	4882428	0	4882428			
Total:	12947104	1343608	10034648			

9: -c

shivank@shivank-Vostro-5568:~\$ free -c 3

	total	used	free	shared	buff/cache	available
--	-------	------	------	--------	------------	-----------

```
Mem:      8064676    1343584    5152004    426296    1569088    6189996
Swap:    4882428         0    4882428
```

```
      total      used      free   shared buff/cache   available
Mem:    8064676    1343376    5147912    430596    1573388    6185904
Swap:   4882428         0    4882428
```

```
      total      used      free   shared buff/cache   available
Mem:    8064676    1343428    5147788    430668    1573460    6185780
Swap:   4882428         0    4882428
```

10: -s

```
shivank@shivank-Vostro-5568:~$ free -c 3 -s 3
```

```
      total      used      free   shared buff/cache   available
Mem:    8064676    1342168    5153400    426160    1569108    6194624
Swap:   4882428         0    4882428
```

```
      total      used      free   shared buff/cache   available
Mem:    8064676    1342032    5153524    426160    1569120    6194756
Swap:   4882428         0    4882428
```

```
      total      used      free   shared buff/cache   available
Mem:    8064676    1342144    5153400    426160    1569132    6194644
Swap:   4882428         0    4882428
```

11: -w

```
shivank@shivank-Vostro-5568:~$ free -w
```

```
      total      used      free   shared  buffers    cache   available
Mem:    8064676    1342064    5153308    426320    102496    1466808    6194596
Swap:   4882428         0    4882428
```

whatis

Description : display one-line manual page descriptions

Syntax \$ **whatis** [-dlv?V] [-r|-w] [-s list] [-m system[,...]] [-M path] [-L locale] [-C file] name

Options

-d, --debug	Print debugging information.
-v, --verbose	Print verbose warning messages.
-r, --regex	Interpret each name as a regular expression.
-w, --wildcard	Interpret each name as a pattern containing shell style wildcards.
-l, --long	Do not trim output to the terminal width.

=> whatis command

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ whatis cat
 cat (1) - concatenate files and print on the standard output

1: -d

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ whatis -d cat
 From the config file /etc/manpath.config:
 Mandatory mandir `/usr/man'.
 Mandatory mandir `/usr/share/man'.
 :

whatis: concatenate files and print on the standard output

cat (1) - concatenate files and print on the standard output
 hashtable_free: 1 entries, 1 (100%) unique

2: -v

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ whatis -v whereis
 whereis (1) - locate the binary, source, and manual page files for a command

3: -r

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ whatis -r cat
 xcb_dri2_authenticate (3) - (unknown subject)
 :
 :
 XtWidgetToApplicationContext (3) - create, destroy, and obtain an application context
 xzcat (1) - Compress or decompress .xz and .lzma files
 zcat (1) - compress or expand files

whereis

Description : Locate the binary, source, and manual-page files for a command.

Syntax \$ whereis [options] [-BMS <dir>... -f] <name>

Options

-b	search only for binaries
-m	search only for manuals and infos
-s	search only for sources
-u	search for unusual entries
-l	output effective lookup paths

=>whereis command

```
shivank@shivank-Vostro-5568:~$ whereis cat
cat: /bin/cat /usr/share/man/man1/cat.1.gz
```

1: -b

```
shivank@shivank-Vostro-5568:~$ whereis -b cat
cat: /bin/cat
```

2: -m

```
shivank@shivank-Vostro-5568:~$ whereis -m cat
cat: /usr/share/man/man1/cat.1.gz
```

3: -l

```
shivank@shivank-Vostro-5568:~$ whereis -l
bin: /usr/bin
bin: /usr/sbin
:
:
:
src: /usr/src/nvidia-384-384.111
src: /usr/src/linux-headers-4.10.0-28
src: /usr/src/linux-headers-4.13.0-32
```

find

Description : Search for files in a directory hierarchy

Syntax \$ find [where to start searching from] [expression determines what to find] [-options] [what to find]

Options

-P	Never follow symbolic links.
-L	Follow symbolic links.
-H	Do not follow symbolic links, except while processing the command line arguments
-name	Search a file with specific name.
-empty	Search for empty files and directories.
-perm	Search for file with entered permissions

=>find command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ find scripts/  
scripts/  
scripts/avgmarks.sh  
scripts/fibonacci.sh  
scripts/code2.sh  
scripts/leapyear.sh  
scripts/factorial.sh  
scripts/code1  
scripts/evenodd.sh  
scripts/pallindrome.sh
```

1: -name

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ find -name mvt  
./ccodes/mvt  
shivank@shivank-Vostro-5568:~/Documents/UE163095$ find -name *.cpp  
./ccodes/mvt.cpp  
./ccodes/sjf_preemptive.cpp  
./ccodes/fcfs.cpp  
./ccodes/RR.cpp  
./ccodes/Banker.cpp  
./ccodes/sjn_non_preemptive.cpp  
./ccodes/threadcode.cpp  
./ccodes/mft.cpp  
./ccodes/Prioritynp.cpp  
./ccodes/Priorityp.cpp
```

2: -empty

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ find -empty  
./Untitled Document
```

3: -perm

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ find -perm 664  
./ccodes/mvt.cpp  
./ccodes/sjf_preemptive.cpp  
./ccodes/fcfs.cpp  
./ccodes/RR.cpp  
./ccodes/Banker.cpp  
./ccodes/sjn_non_preemptive.cpp  
./ccodes/bankerin.txt  
./ccodes/threadcode.cpp  
./ccodes/mft.cpp  
./ccodes/Untitled Document  
./ccodes/Prioritynp.cpp
```

```
./ccodes/Priorityp.cpp
./file2
./count.txt
./myfile1
./myfile
./Untitled Document
```

type

Description : Describe a command, for each name, indicate how it would be interpreted if used as a command name.

Syntax \$ type [-atp] [name ...]

Options

	type prints a single word which is one of: `alias' (shell alias) `function' (shell function) `builtin' (shell builtin) `file' (disk file)
-t	`keyword' (shell reserved word)
-p	type either returns the name of the disk file that would be executed, or nothing if `-t' would not return `file'.
-a	type returns all of the places that contain an executable named file. This includes aliases and functions, if and only if the `-p' option is not also used.

=> type command

```
shivank@shivank-Vostro-5568:~$ type cat
cat is /bin/cat
```

1: -t

```
shivank@shivank-Vostro-5568:~$ type -t cat
file
```

2: -p

```
shivank@shivank-Vostro-5568:~$ type -p cat
/bin/cat
```

3: -a

```
shivank@shivank-Vostro-5568:~$ type -a cat
cat is /bin/cat
```

Practical 5

Commands : sleep, shutdown, ; (semicolon), | (pipe), & (on after the other command), sort, head, tail, more, less, banner, paste

sleep

Description : delay for a specified amount of time

Syntax \$ **sleep** NUMBER[SUFFIX]...
sleep OPTION

Options

Pause for NUMBER seconds. SUFFIX may be 's' for seconds (the default), 'm' for minutes, 'h' for hours or 'd' for days. Unlike most implementations that require NUMBER be an integer, here NUMBER may be an arbitrary floating point number. Given two or more arguments, pause for the amount of time specified by the sum of their values.

=> sleep command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sleep 5s
shivank@shivank-Vostro-5568:~/Documents/UE163095$
```

=>This added a time delay for 5 seconds

shutdown

Description : Halt, power-off or reboot the machine

Syntax \$ **shutdown** [OPTIONS...] [TIME] [WALL...]

Options

-c	Cancel a shutdown that is in progress.
-f	Reboot fast, by suppressing the normal call to fsck when rebooting.
-h	Halt the system when shutdown is complete.
-k	Print the warning message, but suppress actual shutdown.
-n	Perform shutdown without a call to init.
-r	Reboot the system when shutdown is complete.
-t sec	Ensure a sec-second delay between killing processes and changing the runlevel.

=>shutdown command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ shutdown
```

1: -t

shivank@shivank-Vostro-5568:~\$ shutdown -t 20

Shutdown scheduled for Tue 2018-02-13 14:48:57 IST, use 'shutdown -c' to cancel.

2: -c

shivank@shivank-Vostro-5568:~\$ shutdown -c

;(semicolon)

Description : Many commands run by entering all commands at once separated by ;(semicolon)

Syntax \$ **command1; command2; command3**

=> using ;(semicolon)

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ who;whoami;uname

shivank tty7 2018-04-21 12:48 (:0)

shivank

Linux

| (Pipeline)

Description : The pipeline operator takes the output of one command and gives it as the input to other command

Syntax \$ **command1 | command2 | command3**

=>using | (Pipeline)

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ echo -e 'cat\ndog' | grep a

cat

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ echo -e 'cat\ndog' | grep o

dog

=>The output of each command in the pipeline is connected via a pipe to the input of the next command. That is, each command reads the previous command's output. This connection is performed before any redirections specified by the command.

&(ampersand)

Description : The & makes the command run in the background.

Syntax \$ **command &**

=>using &

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ find | grep .cpp &

[1] 6816

shivank@shivank-Vostro-5568:~/Documents/UE163095\$./ccodes/mvt.cpp

./ccodes/sjf_preemptive.cpp

./ccodes/fcfs.cpp

./ccodes/threadcode (copy).cpp

./ccodes/producerconsumer.cpp

./ccodes/RR.cpp

```
./ccodes/Banker.cpp
./ccodes/sjn_non_preemptive.cpp
./ccodes/threadcode.cpp
./ccodes/mft.cpp
./ccodes/Prioritynp.cpp
./ccodes/Priorityp.cpp
```

sort

Description : Write sorted concatenation of all FILE(s) to standard output

Syntax \$ sort [OPTION]... [FILE]...

Options

-c	Check whether the given files are already sorted: if they are not all sorted, print an error message and exit with a status of 1.
-m	Merge the given files by sorting them as a group.
-b	Ignore leading blanks when finding sort keys in each line.
-d	Sort in "phone directory" order: ignore all characters except letters, digits and blanks when sorting.
-f	Fold lowercase characters into the equivalent uppercase characters when sorting so that, for example, `b' and `B' sort as equal.

=>sort command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sort myfile
```

```
abc xyz pqr efg
hello world
      in      file      1
new line
pqr efg abc xyz
xyz abc efg pqr
```

1:-d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sort -d file
```

```
12
23
32
34
355
65
76
97
98
```

2: -g

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sort -g file
```

```
12
23
32
34
65
76
97
98
355
```

3: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sort -g -r file
```

```
355
98
97
76
65
34
32
23
12
```

4: -m

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ sort -m file myfile
```

```
34
65
23
12
76
32
98
355
abc xyz pqr efg
xyz abc efg pqr
pqr efg abc xyz
```

```
hello world
```

```
      in      file      1
new line
```


5: -c

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ sort -c file
 sort: file:3: disorder: 23

head

Description : Print the first 10 lines of each FILE to standard output.

Syntax \$ head [OPTION]... [FILE]...

Options

-c, --bytes=[-]NUM	Print the first NUM bytes of each file; With the leading '-', print all but the last NUM bytes of each file
-n, --lines=[-]NUM	print the first NUM lines instead of the first 10 With the leading '-', print all but the last NUM lines of each file
-q, --quiet, --silent	never print headers giving file names
-v, -v, --verbose	always print headers giving file names
-z, --zero-terminated	line delimiter is NUL, not newline

=>head command

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ head myfile
 hello world
 this is line 2
 the quick brown
 fox jumps over
 the lazy dog
 pallindrome is a string of
 characters

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
 Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,

1: -c

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ head -c 50 myfile
 hello world
 this is line 2
 the quick brown
 fox jumshivank

2: -n

```
@shivank-Vostro-5568:~/Documents/UE163095$ head -n 5 myfile
hello world
this is line 2
the quick brown
fox jumps over
the lazy dog
```

3: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ head -v myfile
==> myfile <==
hello world
this is line 2
the quick brown
fox jumps over
the lazy dog
pallindrome is a string of
characters
```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,

4: -z

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ head -z myfile
hello world
this is line 2
the quick brown
fox jumps over
the lazy dog
pallindrome is a string of
characters
```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum
passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of
Lorem Ipsum.

tail

Description : Print the last 10 lines of each FILE to standard output.

Syntax \$ tail [OPTION]... [FILE]...

Options

-c, --bytes=[+]NUM	Output the last NUM bytes; or use -c +NUM to output starting with byte NUM of each file
-n, --lines=[+]NUM	Output the last NUM lines, instead of the last 10 or use -n +NUM to output starting with line NUM
-q, --quiet, --silent	Never output headers giving file names
-v, -v, --verbose	Always output headers giving file names
-z, --zero-terminated	Line delimiter is NUL, not newline

=>tail command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tail myfile
pallindrome is a string of
characters
```

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

1: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tail -c 300 myfile
centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum
passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of
Lorem Ipsum.
```

2: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tail -n 5 myfile
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
```

It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

3: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tail -v myfile  
==> myfile <==  
pallindrome is a string of  
characters
```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

4: -z

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tail -z myfile  
hello world  
this is line 2  
the quick brown  
fox jumps over  
the lazy dog  
pallindrome is a string of  
characters
```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

more**Description :** File perusal filter for crt viewing**Syntax** \$ more [options] file...**Options**

-d	Prompt with "[Press space to continue, 'q' to quit.]", and display "[Press 'h' for instructions.]" instead of ringing the bell when an illegal key is pressed.
-l	Do not pause after any line containing a ^L (form feed).
-f	Count logical lines, rather than screen lines (i.e., long lines are not folded).
-p	Do not scroll. Instead, clear the whole screen and then display the text. Notice that this option is switched on automatically if the executable is named page.
-c	Do not scroll. Instead, paint each screen from the top, clearing the remainder of each line as it is displayed.
-s	Squeeze multiple blank lines into one.

=>more command

```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
--More--(13%)

```

1: -d

```

but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem I
psum.
hello world
this is line 2
the quick brown
--More--(6%)[Press space to continue, 'q' to quit.]

```

=>The more command is a command line utility for viewing the contents of a file or files once screen at a time. It supports navigating forwards and backwards through a file and is primarily used for viewing the contents of a file. It also supports searching for strings or regular expressions and opening the file at the current point in a text editor.

less

Description : Page through text one screenful at a time, Search through output

Syntax \$ less [options]

Options

-p	pattern : it tells less to start at the first occurrence of pattern in the file
-E	Causes less to automatically exit the first time it reaches end of file.
-F	Causes less to exit if entire file can be displayed on first screen
-N	It will show output along with line numbers
-s	Causes consecutive blank lines to be squeezed into a single blank line

=>less command

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ less myfile2

```
this is line 2
the quick brown
fox jumps over
the lazy dog
pallindrome is a string of
characters
:
```

```
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem I
psum.
(END)
```

1: -p

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ less -p "hello" myfile2

```
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem I
psum.
hello world
this is line 2
the quick brown
myfile2
```

2: -F

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ less -F myfile
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ less -F myfile2
```

```

Lorem Ipsum is simply dummy text of the printing and typesetting industry.
Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
when an unknown printer took a galley of type and scrambled it to make a type specimen book.
It has survived not only five centuries,
but also the leap into electronic typesetting, remaining essentially unchanged.
It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages,
and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem I
psum.
hello world
this is line 2
the quick brown
myfile2

```

3: -N

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ less -N myfile
```

```

 1 hello world
 2 this is line 2
 3 the quick brown
 4 fox jumps over
 5 the lazy dog
 6 pallindrome is a string of
 7 characters
 8
 9 Lorem Ipsum is simply dummy text of the printing and typesetting industry.
10 Lorem Ipsum has been the industry's standard dummy text ever since the 1500s,
11 when an unknown printer took a galley of type and scrambled it to make a type specimen book.
12 It has survived not only five centuries,
13 but also the leap into electronic typesetting, remaining essentially unchanged.
14 It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum pa
14 ssages,
15 and more recently with desktop publishing software like Aldus PageMaker including versions of
15 Lorem Ipsum.
myfile (END)

```

banner**Description : Print large banner****Syntax \$ banner text**

```
shivank@shivank-Vostro-5568:~$ banner Shivank
```

```

#####
#   # #   #   #   #   #   ##   #   #   #   #
#       #   #   #   #   #   #   ##   #   #   #
#####  #####   #   #   #   #   #   #   #   #####
      #   #   #   #   #   #   #####   #   #   #   #
#   #   #   #   #   #   #   #   #   #   ##   #   #
#####   #   #   #   ##   #   #   #   #   #   #

```

paste

Description : Write lines consisting of the sequentially corresponding lines from each FILE, separated by TABs, to standard output.

Syntax `$ paste [OPTION]... [FILE]...`

Options

-d, --delimiters=LIST	Reuse characters from LIST instead of TABs
-s, --serial	Paste one file at a time instead of in parallel
-z, --zero-terminated	Line delimiter is NUL, not newline

=>paste command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ paste state capital
Arunachal Pradesh  Itanagar
Assam Dispur
Andhra Pradesh     Hyderabad
Bihar Patna
Chhattisgarh Raipur
Punjab chandigarh
```

1: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ paste -d "-" state capital
Arunachal Pradesh-Itanagar
Assam-Dispur
Andhra Pradesh-Hyderabad
Bihar-Patna
Chhattisgarh-Raipur
Punjab-chandigarh
shivank@shivank-Vostro-5568:~/Documents/UE163095$ paste -d "\n" state capital
Arunachal Pradesh
Itanagar
Assam
Dispur
Andhra Pradesh
Hyderabad
Bihar
Patna
Chhattisgarh
Raipur
Punjab
chandigarh
```


2: -z

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ paste -z state capital

Arunachal Pradesh

Assam

Andhra Pradesh

Bihar

Chhattisgarh

Punjab

Itanagar

Dispur

Hyderabad

Patna

Raipur

chandigarh

3: -s

shivank@shivank-Vostro-5568:~/Documents/UE163095\$ paste -s state capital

Arunachal Pradesh Assam Andhra Pradesh Bihar Chhattisgarh Punjab

Itanagar Dispur Hyderabad Patna Raipur chandigarh

Practical 6

Commands : adduser, useradd, deluser, userdel, umask, chmod, cut, cal, ulimit, clear, finger, wall, write

adduser

Description : Add a normal user

Syntax \$ adduser [options] [--home DIR] [--shell SHELL] [--no-create-home] [--uid ID] [--firstuid ID] [--lastuid ID] [--ingroup GROUP | --gid ID] [--disabled-password] [--disabled-login] [--gecos GECOS] [--add_extra_groups] [--encrypt-home] user

Options

--quiet	Suppress informational messages, only show warnings and errors.
--no-create-home	Create a system user or group.
--system	Do not create the home directory, even if it doesn't exist.

=>adduser command

```
shivank@shivank-Vostro-5568:~$ sudo adduser sb
[sudo] password for shivank:
Adding user `sb' ...
Adding new group `sb' (1001) ...
Adding new user `sb' (1001) with group `sb' ...
Creating home directory `/home/sb' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for sb
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n] y
```

1: --quiet

```
shivank@shivank-Vostro-5568:~$ sudo adduser --quiet sb
```

```
Enter new UNIX password:
```

```
Retype new UNIX password:
```

```
passwd: password updated successfully
```

```
Changing the user information for sb
```

```
Enter the new value, or press ENTER for the default
```

```
Full Name []:
```

```
Room Number []:
```

```
Work Phone []:
```

```
Home Phone []:
```

```
Other []:
```

```
Is the information correct? [Y/n] y
```

2: --no-create-home

```
shivank@shivank-Vostro-5568:~$ sudo adduser --no-create-home sb
```

```
Adding user `sb' ...
```

```
Adding new group `sb' (1001) ...
```

```
Adding new user `sb' (1001) with group `sb' ...
```

```
Not creating home directory `/home/sb'.
```

```
Enter new UNIX password:
```

```
Retype new UNIX password:
```

```
passwd: password updated successfully
```

```
Changing the user information for sb
```

```
Enter the new value, or press ENTER for the default
```

```
Full Name []:
```

```
Room Number []:
```

```
Work Phone []:
```

```
Home Phone []:
```

```
Other []:
```

```
Is the information correct? [Y/n] y
```

3: --system

```
shivank@shivank-Vostro-5568:~$ sudo adduser --system ab
```

```
Adding system user `ab' (UID 122) ...
```

```
Adding new user `ab' (UID 122) with group `nogroup' ...
```

```
Creating home directory `/home/ab' ...
```

deluser**Description :** remove a normal user from the system**Syntax** `$ deluser [options] [--force] [--remove-home] [--remove-all-files] [--backup] [--backup-to DIR] user`**Options**

--remove-home	remove the users home directory and mail spool
--system	only remove if system user
--quiet, -q	don't give process information to stdout
--remove-all-files	remove all files owned by user
--backup	backup files before removing.

=>deluser command

```
shivank@shivank-Vostro-5568:~$ sudo deluser sb
```

```
Removing user `sb' ...
```

```
Warning: group `sb' has no more members.
```

```
Done.
```

1: --remove-home

```
shivank@shivank-Vostro-5568:~$ sudo deluser --remove-home sb
```

```
Looking for files to backup/remove ...
```

```
Removing files ...
```

```
Removing user `sb' ...
```

```
Warning: group `sb' has no more members.
```

```
Done.
```

2: --system

```
shivank@shivank-Vostro-5568:~$ sudo deluser --system sb
```

```
The user `sb' is not a system user. Exiting.
```

3: -q

```
shivank@shivank-Vostro-5568:~$ sudo deluser -q sb
```

```
shivank@shivank-Vostro-5568:~$
```

useradd**Description :** create a new user or update default new user information**Syntax** \$ useradd [options] LOGIN**Options**

-b, --base-dir BASE_DIR	The default base directory for the system if -d HOME_DIR is not specified.
-c, --comment COMMENT	Any text string.
-d, --home-dir HOME_DIR	The new user will be created using HOME_DIR as the value for the user's login directory.
-r, --system	Create a system account.
-s, --shell SHELL	The name of the user's login shell.

=>useradd command

shivank@shivank-Vostro-5568:~\$ sudo useradd sb

shivank@shivank-Vostro-5568:~\$

1: -r

shivank@shivank-Vostro-5568:~\$ sudo useradd -r sb

shivank@shivank-Vostro-5568:~\$

userdel**Description :** Delete a user account and related files**Syntax** \$ userdel [options] LOGIN**Options**

-f, --force	This option forces the removal of the user account, even if the user is still logged in.
-R, --root CHROOT_DIR	Apply changes in the CHROOT_DIR directory and use the configuration files from the CHROOT_DIR directory.
-Z, --selinux-user	Remove any SELinux user mapping for the user's login.
-r, --remove	Files in the user's home directory will be removed along with the home directory itself and the user's mail pool.

=> userdel command

```
shivank@shivank-Vostro-5568:~$ sudo userdel sb
shivank@shivank-Vostro-5568:~$
```

1: -f

```
shivank@shivank-Vostro-5568:~$ sudo userdel -f sb
shivank@shivank-Vostro-5568:~$
```

2: -r

```
shivank@shivank-Vostro-5568:~$ sudo userdel -r sb
userdel: sb mail spool (/var/mail/sb) not found
userdel: sb home directory (/home/sb) not found
shivank@shivank-Vostro-5568:~$
```

umask

Description : set file mode creation mask

Syntax \$ umask [-S] [*mask*]

Options

-S	Accept a symbolic representation of a <i>mask</i> , or return one
mask	If a valid <i>mask</i> is specified, the umask is set to this value. If no <i>mask</i> is specified, the current umask value is returned.

=> umask command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask
0002
```

1: mask

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask 234
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask
0234
```

2: -S

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask -S
u=rX,g=r,o=wx
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask 002
shivank@shivank-Vostro-5568:~/Documents/UE163095$ umask -S
u=rwx,g=rwx,o=rx
```

chmod

Description : Change the mode of each FILE to MODE.

Syntax \$ chmod [Reference][Operator][Mode] file...

Options

Reference	Class	Description
u	owner	File's owner
g	group	Users who are members of the file's group
o	others	Users who are neither owner nor member of the group
a	all	All three of the above, same as ugo

Operator	Description
+	Add the specified mode to the specified classes
-	Remove the specified mode to the specified classes
=	The modes specified are to be made the exact modes for the specified classes

=>chmod command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ cat b.py
```

```
import numpy
```

```
i=1
```

```
j=1
```

```
for i in range(1,10):
```

```
    j=i*j
```

```
print(j)
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 60 Apr 22 20:01 b.py
```

1: +

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ chmod u+x b.py
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ ls -l
```

```
total 4
```

```
-rwxrw-r-- 1 shivank shivank 60 Apr 22 20:01 b.py
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ chmod a+x b.py
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ ls -l
```

```
total 4
```

```
-rwxrwxr-x 1 shivank shivank 60 Apr 22 20:01 b.py
```

2: -

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ chmod a-x b.py
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ ls -l
```

```
total 4
```

```
-rw-rw-r-- 1 shivank shivank 60 Apr 22 20:01 b.py
```

3: =

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ chmod a=rw b.py
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ ls -l
```

```
total 4
```

```
-rw-rw-rw- 1 shivank shivank 60 Apr 22 20:01 b.py
```

cut

Description : Divide a file into several parts (columns)

Syntax \$ cut [OPTION]... [FILE]...

Options

-b BYTE-LIST or --bytes=BYTE-LIST	Print only the bytes in positions listed in BYTE-LIST. Tabs and backspaces are treated like any other character; they take up 1 byte
-c CHARACTER-LIST or --characters=CHARACTER-LIST	Print only characters in positions listed in CHARACTER-LIST.
-f FIELD-LIST or --fields=FIELD-LIST	Print only the fields listed in FIELD-LIST. Fields are separated by a TAB character by default.
-d INPUT_DELIM_BYTE or --delimiter=INPUT_DELIM_BYTE	For '-f', fields are separated in the input by the first character in INPUT_DELIM_BYTE (default is TAB).
-n	Do not split multi-byte characters (no-op for now).
-s or --only-delimited	For '-f', do not print lines that do not contain the field separator character.
--output-delimiter=OUTPUT_DELIM_STRING	For '-f', output fields are separated by OUTPUT_DELIM_STRING. The default is to use the input delimiter.

=>cut command**1: -b**

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello world' | cut -b 5,8
oo
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello world' | cut -b 5,8,10
ool
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello world' | cut -b 5-10
o worl
```

2: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ echo 'hello: world!' | cut -c 2-10
ello: wor
```

3: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cut -d ' ' -f 1,3 myfile
abc pqr
xyz efg
pqr abc
```

```
hello
```

```
      in      file      1
new
```

4: -f

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ cut -d ' ' -f 2 myfile
xyz
abc
efg
```

```
world
```

```
      in      file      1
line
```

cal**Description :** Displays a calendar and the date of Easter**Syntax \$****Options**

-h	Turns off highlighting of today.
-j	Display Julian days (days one-based, numbered from January 1).
-y	Display a calendar for the specified year. This option is implied when a year but no month are specified on the command line.
-3	Display the previous, current and next month surrounding today.
-1	Display only the current month. This is the default.
-C	Switch to cal mode.
-N	Switch to ncal mode.

=>cal command

```
shivank@shivank-Vostro-5568:~$ cal
  April 2018
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30
```

1: -h

```
shivank@shivank-Vostro-5568:~$ cal -h
  April 2018
Su Mo Tu We Th Fr Sa
 1  2  3  4  5  6  7
 8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30
```

2: -j

```
shivank@shivank-Vostro-5568:~$ cal -j
      April 2018
Su Mo Tu We Th Fr Sa
 91 92 93 94 95 96 97
 98 99 100 101 102 103 104
105 106 107 108 109 110 111
112 113 114 115 116 117 118
119 120
```

3: -y

```
shivank@shivank-Vostro-5568:~$ cal -y
      2018
      January      February      March
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
  1  2  3  4  5  6    1  2  3    1  2  3
  7  8  9 10 11 12 13    4  5  6  7  8  9 10    4  5  6  7  8  9 10
14 15 16 17 18 19 20   11 12 13 14 15 16 17   11 12 13 14 15 16 17
21 22 23 24 25 26 27   18 19 20 21 22 23 24   18 19 20 21 22 23 24
28 29 30 31           25 26 27 28           25 26 27 28 29 30 31

      April      May      June
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
  1  2  3  4  5  6  7    1  2  3  4  5    1  2
  8  9 10 11 12 13 14    6  7  8  9 10 11 12    3  4  5  6  7  8  9
15 16 17 18 19 20 21   13 14 15 16 17 18 19   10 11 12 13 14 15 16
22 23 24 25 26 27 28   20 21 22 23 24 25 26   17 18 19 20 21 22 23
29 30           27 28 29 30 31           24 25 26 27 28 29 30

      July      August      September
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
  1  2  3  4  5  6  7    1  2  3  4    1
  8  9 10 11 12 13 14    5  6  7  8  9 10 11    2  3  4  5  6  7  8
15 16 17 18 19 20 21   12 13 14 15 16 17 18    9 10 11 12 13 14 15
22 23 24 25 26 27 28   19 20 21 22 23 24 25   16 17 18 19 20 21 22
29 30 31           26 27 28 29 30 31           23 24 25 26 27 28 29
                                     30

      October      November      December
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
  1  2  3  4  5  6    1  2  3    1
  7  8  9 10 11 12 13    4  5  6  7  8  9 10    2  3  4  5  6  7  8
14 15 16 17 18 19 20   11 12 13 14 15 16 17    9 10 11 12 13 14 15
21 22 23 24 25 26 27   18 19 20 21 22 23 24   16 17 18 19 20 21 22
28 29 30 31           25 26 27 28 29 30   23 24 25 26 27 28 29
                                     30 31
```

4: -3

```
shivank@shivank-Vostro-5568:~$ cal -3
          30 31
March 2018      April 2018      May 2018
Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa Su Mo Tu We Th Fr Sa
 1 2 3         1 2 3 4 5 6 7         1 2 3 4 5
 4 5 6 7 8 9 10 8 9 10 11 12 13 14 6 7 8 9 10 11 12
11 12 13 14 15 16 17 15 16 17 18 19 20 21 13 14 15 16 17 18 19
18 19 20 21 22 23 24 22 23 24 25 26 27 28 20 21 22 23 24 25 26
25 26 27 28 29 30 31 29 30      27 28 29 30 31
```

5: -N

```
shivank@shivank-Vostro-5568:~$ cal -N
April 2018
Su 1 8 15 22 29
Mo 2 9 16 23 30
Tu 3 10 17 24
We 4 11 18 25
Th 5 12 19 26
Fr 6 13 20 27
Sa 7 14 21 28
```

ulimit

Description : The ulimit command sets or reports user process resource limits.

Syntax \$ ulimit [option] [*Limit*]

Options

-a	Lists all of the current resource limits
-c	Specifies the size of core dumps, in number of 512-byte blocks
-d	Specifies the size of the data area, in number of K bytes
-f	Sets the file size limit in blocks when the Limit parameter is used, or reports the file size limit if no parameter is specified. The -f flag is the default
-H	Specifies that the hard limit for the given resource is set. If you have root user authority, you can increase the hard limit. Anyone can decrease it
-m	Specifies the size of physical memory, in number of K bytes
-n	Specifies the limit on the number of file descriptors a process may have
-s	Specifies the stack size, in number of K bytes
-S	Specifies that the soft limit for the given resource is set. A soft limit can be increased up to the value of the hard limit. If neither the -H nor -S flags are specified, the limit applies to both
-t	Specifies the number of seconds to be used by each process

=> Ulimit command

```
shivank@shivank-Vostro-5568:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
file size               (blocks, -f) unlimited
pending signals         (-i) 30825
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority      (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes      (-u) 30825
virtual memory          (kbytes, -v) unlimited
file locks              (-x) unlimited
```

=>The default limits are defined and applied when a new user is added to the system. Limits are categorized as either soft or hard.

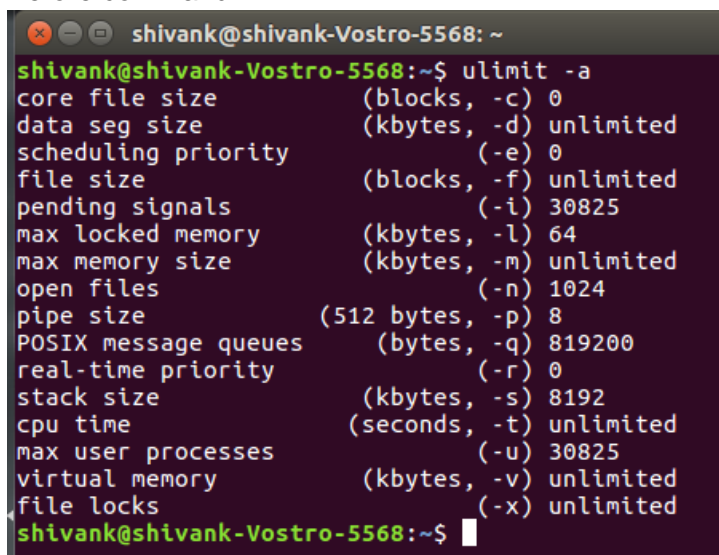
clear

Description : clear the terminal screen

Syntax \$ clear

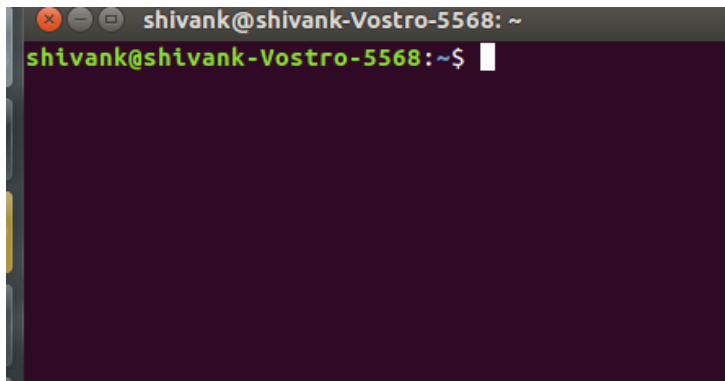
=>clear command

Before command



```
shivank@shivank-Vostro-5568:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
file size               (blocks, -f) unlimited
pending signals         (-i) 30825
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority      (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes      (-u) 30825
virtual memory          (kbytes, -v) unlimited
file locks              (-x) unlimited
shivank@shivank-Vostro-5568:~$
```

After clear command



finger

Description : User information lookup program

Syntax \$ finger [-lmsp] [user ...] [user@host ...]

Options

-l	Force long output format.
-m	Match arguments only on user name (not first or last name).
-p	Suppress printing of the .plan file in a long format printout.
-s	Force short output format.

=> finger command

shivank@shivank-Vostro-5568:~\$ finger

```

Login   Name    Tty   Idle Login Time  Office   Office Phone
shivank Shivank  tty7   1d Apr 29 14:07 (:0)

```

1: -s

shivank@shivank-Vostro-5568:~\$ finger shivank

```

Login: shivank                Name: Shivank
Directory: /home/shivank      Shell: /bin/bash
On since Sun Apr 29 14:07 (IST) on tty7 from :0
    1 day 2 hours idle

```

No mail.

No Plan.

shivank@shivank-Vostro-5568:~\$ finger -s shivank

```

Login   Name    Tty   Idle Login Time  Office   Office Phone
shivank Shivank  tty7   1d Apr 29 14:07 (:0)

```

2: -l

```
shivank@shivank-Vostro-5568:~$ finger -l
Login: shivank          Name: Shivank
Directory: /home/shivank  Shell: /bin/bash
On since Sun Apr 29 14:07 (IST) on tty7 from :0
    1 day 2 hours idle
No mail.
No Plan.
```

wall

Description : Write a message to all users

Syntax \$ wall [-n] [-t timeout] [message | file]

Options

-n, --nobanner	Suppress the banner.
-t, --timeout timeout	Abandon the write attempt to the terminals after timeout seconds. This timeout must be a positive integer. The default value is 300 seconds, which is a legacy from the time when people ran terminals over modem lines.

=> wall command

```
shivank@shivank-Vostro-5568:~$ wall there is hope
```

=>On the other terminal message was displayed as:

```
Broadcast message from shivank (pts/1) (Thu May 3 08:28:21 2018):
there is hope
```

write

Description : send a message to another user

Syntax \$ write user [tty]

=> write command

```
shivank@shivank-Vostro-5568:~$ write Shivank pts/7
```

```
Message from root@wiki on pts/8 at 11:19 ...
test
```

Practical 7

Commands : gzip, gunzip, zip, unzip, tar, split, ps, kill, top, nice, nohup, batch, at, crontab

gzip

Description : Compress or uncompress FILEs (by default, compress FILEs in-place).

Syntax \$ gzip [OPTION]... [FILE]...

Options

-c, --stdout	Write on standard output, keep original files unchanged
-d, --decompress	Decompress
-k, --keep	Keep (don't delete) input files
-l, --list	List compressed file contents
-q, --quiet	Suppress all warnings
-r, --recursive	Operate recursively on directories

=>gzip command

1: -k

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gzip -k myfile myfile1
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
```

```
a      ccodes   Disk Sheduling  myfile  myfile1.gz  myfile.gz  state
capital count.txt file2          myfile1 myfile2   scripts
```

2: -c

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gzip -c myfile
```

```
%ZmyfileUQn0
```

```
(x: ,E5>?)> _m.5.mA _
```

```
3F-+U<]~Tuu{?hu*.Tl rW
```

```
M~sy* m iH% t u@'.@['py!;
```

```
+!}ش'.
```

```
≡Y}ejn&rE`fR_]#/  
g^
```

```
8~^N8Vo@~H.fp5GV%rU@4W^"EWv  
>+=1VH8H7fScũe$!@H]X&JcU.;\bG TW  
Hu;:;<M$V~shivank@shivank-Vostro-5568:~/Documents/UE163095$
```


3: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gzip -l myfile.gz
      compressed      uncompressed  ratio uncompressed_name
          419           686  42.6% myfile
```

4: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls a/
b.py
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gzip -r -k a
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls a/
b.py b.py.gz
```

5: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gzip -r -d a
gzip: a/b.py already exists; do you wish to overwrite (y or n)? y
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls a/
b.py
```

gunzip

Description : Uncompress FILEs (by default, in-place).

Syntax \$ **gunzip** [OPTION]... [FILE]...

Options

-c, --stdout	Write on standard output, keep original files unchanged
-k, --keep	Keep (don't delete) input files
-l, --list	List compressed file contents
-q, --quiet	Suppress all warnings
-r, --recursive	Operate recursively on directories

=>gunzip command**1: -c**

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gunzip -c myfile.gz
hello world
this is line 2
the quick brown
fox jumps over
the lazy dog
palindrome is a string of characters
```

2: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ gunzip -l myfile.gz
      compressed      uncompressed  ratio uncompressed_name
         419           686  42.6% myfile
```

zip

Description : The default action is to add or replace zipfile entries from list, which can include the special name - to compress standard input.

Syntax \$ zip [-options] [-b path] [-t mmddyyyy] [-n suffixes] [zipfile list] [-xi list]

Options

-u	Update: only changed or new files
-d	Delete entries in zipfile
-r	Recurse into directories
-m	Move into zipfile (delete OS files)
-x	Exclude the following names
-v	Verbose operation/print version info

=>zip command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip abc.zip myfile myfile1 myfile2
adding: myfile (deflated 43%)
adding: myfile1 (deflated 94%)
adding: myfile2 (deflated 98%)
```

1: -u

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip -u abc.zip file2
adding: file2 (stored 0%)
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zipinfo -1 abc.zip
myfile
myfile1
myfile2
file2
```

2: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip -r a.zip a
adding: a/ (stored 0%)
adding: a/b.py (stored 0%)
```

3: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip -d abc.zip file2
deleting: file2
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zipinfo -1 abc.zip
myfile
myfile1
myfile2
```

4: -m

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a      a.zip  ccodes myfile myfile2 state
abc.zip capital count.txt file2  myfile1 scripts
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip -m file.zip myfile myfile1 myfile2
adding: myfile (deflated 43%)
adding: myfile1 (deflated 94%)
adding: myfile2 (deflated 98%)
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
a abc.zip a.zip capital ccodes count.txt file2 file.zip scripts state
shivank@shivank-Vostro-5568:~/Documents/UE163095$
```

5: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ zip -v -m file.zip myfile myfile1 myfile2
adding: myfile      (in=686) (out=394) (deflated 43%)
adding: myfile1     (in=8995) (out=507) (deflated 94%)
adding: myfile2     (in=35980) (out=678) (deflated 98%)
total bytes=45661, compressed=1579 -> 97% savings
zip diagnostic: deleting file myfile
zip diagnostic: deleting file myfile1
zip diagnostic: deleting file myfile2
```

unzip

Description : Default action is to extract files in list, except those in xlist, to exdir;

Syntax \$ **unzip [-Z] [-opts[modifiers]] file[.zip] [list] [-x xlist] [-d exdir]**

Options

-p	Extract files to pipe, no messages
-l	List files (short format)
-d	Extract files into exdir
-f	Freshen existing files, create none
-u	Update files, create if necessary

-v	List verbosely/show version info
-x	Exclude files that follow (in xlist)
-n	Never overwrite existing files
-q	Quiet mode (-qq => quieter)
-o	Overwrite files WITHOUT prompting

=>unzip command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip a.zip
```

```
Archive: a.zip
```

```
replace a/b.py? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
```

```
extracting: a/b.py
```

1: -p

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip -p a.zip
```

```
import numpy
```

```
i=1
```

```
j=1
```

```
for i in range(1,10):
```

```
    j=i*j
```

```
print(j)
```

```
shivank@shivan
```

2: -u

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip -u file.zip
```

```
Archive: file.zip
```

3: -v

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip -u -v file.zip
```

```
Archive: file.zip
```

```

Length  Method  Size Cmpr  Date   Time    CRC-32  Name
-----  -
  686 Defl:N   394 43% 2018-04-21 17:21 de145624 myfile
 8995 Defl:N   507 94% 2018-04-21 18:29 c31c6856 myfile1
35980 Defl:N   678 98% 2018-04-21 18:30 e4519175 myfile2
-----  -
 45661      1579 97%                3 files
shivank@shivank-Vostro-5568:~/Documents/UE163095$
```

4: -x

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip -u -v file.zip -x myfile
```

```
Archive: file.zip
```

```
Length Method Size Cmpr Date Time CRC-32 Name
-----
 8995 Defl:N  507 94% 2018-04-21 18:29 c31c6856 myfile1
35980 Defl:N  678 98% 2018-04-21 18:30 e4519175 myfile2
-----
44975      1185 97%                2 files
shivank@shivank-Vostro-5568:~/Documents/UE163095$
```

5: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ unzip -l abc.zip
```

```
Archive: abc.zip
```

```
Length Date Time Name
-----
 686 2018-04-21 17:21 myfile
 8995 2018-04-21 18:29 myfile1
35980 2018-04-21 18:30 myfile2
-----
45661      3 files
```

tar

Description : GNU 'tar' saves many files together into a single tape or disk archive, and can restore individual files from the archive.

Syntax \$ tar [OPTION...] [FILE]...

Options

-A, --catenate, --concatenate	Append tar files to an archive
-c, --create	Create a new archive
-d, --diff, --compare	Find differences between archive and file system
-f, --file=ARCHIVE	Use archive file or device ARCHIVE
-r, --append	Append files to the end of an archive
-t, --list	List the contents of an archive
-u, --update	Only append files newer than copy in archive
-x, --extract, --get	Extract files from an archive

=>tar command**1: -c**

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -cf new.tar myfile myfile1 myfile2
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ ls
```

```
a      a.zip  ccodes  Disk Sheduling  file.zip  myfile1  new.tar  state
```

```
abc.zip  capital  count.txt  file2      myfile  myfile2  scripts
```

2: -t

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -tf new.tar
```

```
myfile
```

```
myfile1
```

```
myfile2
```

3: -u

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -uf new.tar file2
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -tf new.tar
```

```
myfile
```

```
myfile1
```

```
myfile2
```

```
file2
```

4: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -rf new.tar file2
```

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ tar -tf new.tar
```

```
myfile
```

```
myfile1
```

```
myfile2
```

```
file2
```

split

Description : Output pieces of FILE to PREFIXaa, PREFIXab, ...;

Default size is 1000 lines, and default PREFIX is 'x'.

Syntax \$ split [OPTION]... [FILE [PREFIX]]

Options

-a, --suffix-length=N	Generate suffixes of length N (default 2)
-b, --bytes=SIZE	Put SIZE bytes per output file
-l, --lines=NUMBER	Put NUMBER lines/records per output file
-t, --separator=SEP	Use SEP instead of newline as the record separator. '\0' (zero) specifies the NUL character

-d	Use numeric suffixes starting at 0, not alphabetic
-u, --unbuffered	Immediately copy input to output with '-n r/...'
--verbose	Print a diagnostic just before each

=>split command

1: -l

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ split -l 100 myfile2
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ls
myfile2 xaa xab xac xad xae xaf xag xah xai
```

2: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ split -l 100 -a4 myfile2
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ls
myfile2 xaaaa xaaab xaaac xaaad xaaae xaaaf xaaag xaaah xaaai
```

3: -b

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ split -b 200 myfile2
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ls
myfile2 xaj xat xbd xbn xbx xch xcr xdb xdl xdv xef xep xez xfi xft xgd xgn ngx
xaa xak xau xbe xbo xby xci xcs xdc xdm xdw xeg xeq xfa xfk xfu xge xgo
xab xal xav xbf xbp xbz xcj xct xdd xdn xdx xeh xer xfb xfl xfv xgf xgp
xac xam xaw xbg xbg xca xck xcu xde xdo xdy xei xes xfc xfm xfw xgg xgq
xad xan xax xbh xbr xcb xcl xcv xdf xdp xdz xej xet xfd xfn xfx xgh xgr
xae xao xay xbi xbs xcc xcm xcw xdg xdq xea xek xeu xfe xfo xfy xgi xgs
xaf xap xaz xbj xbt xcd xcn xcw xdh xdr xeb xel xev xff xfp xgz xgt
xag xaq xba xbk xbu xce xco xcy xdi xds xec xem xew xfg xfq xga xgk xgu
xah xar xbb xbl xbv xcf xcp xcz xdj xdt xed xen xex xfh xfr xgb xgl xgv
xai xas xbc xbm xbw xcg xcq xda xdk xdu xee xeo xey xfi xfs xgc xgm xgw
```

ps

Description : Report a snapshot of the current processes.

Syntax \$ ps [options]

Options

-A	Select all processes.
-a	Select all processes except both session leaders (see getsid(2)) and processes not associated with a terminal.
-d	Select all processes except session leaders.

-r	Restrict the selection to only running processes.
-T	Select all processes associated with this terminal. Identical to the t option without any argument
-x	Lift the BSD-style "must have a tty" restriction, which is imposed upon the set of all processes when some BSD-style (without "-") options are used or when the ps personality setting is BSD-like. The set of processes selected in this manner is in addition to the set of processes selected by other means. An alternate description is that this option causes ps to list all processes owned by you (same EUID as ps), or to list all processes when used together with the a option.

=>ps command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps
  PID TTY          TIME CMD
 2711 pts/4    00:00:01 bash
10266 pts/4    00:00:00 ps
```

1: -a

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps -a
  PID TTY          TIME CMD
10091 pts/11    00:00:00 man
10103 pts/11    00:00:00 pager
10268 pts/4    00:00:00 ps
```

2: -d

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps -d
  PID TTY          TIME CMD
   2 ?        00:00:00 kthreadd
   4 ?        00:00:00 kworker/0:0H
:
:
10267 ?        00:00:00 kworker/2:4
10269 pts/4    00:00:00 ps
```

3: -T

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps -T
  PID SPID TTY          TIME CMD
 2711 2711 pts/4    00:00:01 bash
10315 10315 pts/4    00:00:00 ps
```


4: -r

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps -r
```

```
PID TTY    STAT  TIME COMMAND
```

```
10316 pts/4  R+    0:00 ps -r
```

5: -x

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/b$ ps -x
```

```
PID TTY    STAT  TIME COMMAND
```

```
1345 ?      Ss    0:00 /lib/systemd/systemd --user
```

```
1346 ?      S     0:00 (sd-pam)
```

```
:
```

```
:
```

```
10103 pts/11  S+    0:00 pager
```

```
10318 pts/4  R+    0:00 ps -x
```

kill

Description : Send a signal to a process

Syntax \$ kill [options] <pid> [...]

Options

-s <signal> --signal <signal>	Specify the signal to be sent. The signal can be specified by using name or number.
-l, --list [signal]	List signal names. This option has optional argument, which will convert signal number to signal name, or other way round.
-L, --table	List signal names in a nice table.
kill -9 -1	Kill all processes you can kill.

=>kill command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095$ kill 12682
```

=>process with pid 12682 was stopped after this command

top**Description :** Display Linux processes**Syntax** \$ top -hv|-bcHiOSs -d secs -n max -u|U user -p pid -o fld -w [cols]**Options**

-b	Batch-mode operation
-d	Delay-time interval as: -d ss.t (secs.tenths)
-n	Number-of-iterations limit as: -n number
-s	Secure-mode operation

=>top command

```

shivank@shivank-Vostro-5568: ~/Documents/UE163095
top - 11:58:41 up 5:47, 1 user, load average: 1.73, 1.40, 1.18
Tasks: 222 total, 1 running, 221 sleeping, 0 stopped, 0 zombie
%Cpu(s): 16.2 us, 3.8 sy, 0.0 ni, 77.5 id, 2.2 wa, 0.0 hi, 0.2 si, 0.0 st
KiB Mem : 8064676 total, 3755616 free, 2177652 used, 2131408 buff/cache
KiB Swap: 4882428 total, 4882428 free, 0 used. 5378796 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM    TIME+  COMMAND
 12535 shivank   20   0 1299788 264312 90904 S   12.5   3.3   3:15.21 chrome
 10812 root       20   0 482260 101540 77136 S    6.2   1.3   0:59.94 Xorg
 13675 shivank   20   0 41940 3692 3064 R    6.2   0.0   0:00.01 top
    1 root       20   0 185648 6212 3936 S    0.0   0.1   0:02.14 systemd
    2 root       20   0      0      0      0 S    0.0   0.0   0:00.01 kthreadd
    4 root       0 -20      0      0      0 S    0.0   0.0   0:00.00 kworker/0:0H
    6 root       0 -20      0      0      0 S    0.0   0.0   0:00.00 mm_percpu_wq
    7 root       20   0      0      0      0 S    0.0   0.0   0:00.19 ksoftirqd/0
    8 root       20   0      0      0      0 S    0.0   0.0   0:13.00 rcu_sched
    9 root       20   0      0      0      0 S    0.0   0.0   0:00.00 rcu_bh
   10 root       rt    0      0      0      0 S    0.0   0.0   0:00.02 migration/0
   11 root       rt    0      0      0      0 S    0.0   0.0   0:00.04 watchdog/0
   12 root       20   0      0      0      0 S    0.0   0.0   0:00.00 cpuhp/0
   13 root       20   0      0      0      0 S    0.0   0.0   0:00.00 cpuhp/1
   14 root       rt    0      0      0      0 S    0.0   0.0   0:00.04 watchdog/1
   15 root       rt    0      0      0      0 S    0.0   0.0   0:00.01 migration/1
   16 root       20   0      0      0      0 S    0.0   0.0   0:00.34 ksoftirqd/1
   18 root       0 -20      0      0      0 S    0.0   0.0   0:00.00 kworker/1:0H
   19 root       20   0      0      0      0 S    0.0   0.0   0:00.00 cpuhp/2
   20 root       rt    0      0      0      0 S    0.0   0.0   0:00.05 watchdog/2
   21 root       rt    0      0      0      0 S    0.0   0.0   0:00.02 migration/2
   22 root       20   0      0      0      0 S    0.0   0.0   0:00.42 ksoftirqd/2
   24 root       0 -20      0      0      0 S    0.0   0.0   0:00.00 kworker/2:0H

```

nice**Description :** Run a program with modified scheduling priority**Syntax** \$ **nice** [**OPTION**] [**COMMAND** [**ARG**]...]**Options**

-n, --adjustment=N	Add integer N to the niceness (default 10)
--help	Display this help and exit
--version	Output version information and exit

=>nice command

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ nice python b.py
362880
```

1: -n

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ cat b.py
import numpy
i=1
j=1
for i in range(1,10):
    j=i*j
print(j)
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ nice -n 20 python b.py
362880
shivank@shivank-Vostro-5568:~/Documents/UE163095/a$ nice -n 0 python b.py
362880
```

nohup**Description :** Run a command immune to hangups, with output to a non-tty**Syntax** \$ **nohup** **COMMAND** [**ARG**]...**nohup** **OPTION****Options**

--help	Display this help and exit
--version	Output version information and exit

=>nohup command

```
shivank@shivank-Vostro-5568:~$ who
shivank  tty7      2018-05-06 11:19 (:0)
shivank@shivank-Vostro-5568:~$ nohup who
nohup: ignoring input and appending output to 'nohup.out'
shivank@shivank-Vostro-5568:~$ cat nohup.out
shivank  tty7      2018-05-06 11:19 (:0)
```

at

Description : at, atq, atrm - queue, examine or delete jobs for later execution

Syntax \$ at [-V] [-q queue] [-f file] [-mMlv] timespec...

at [-V] [-q queue] [-f file] [-mMkv] [-t time]

at -c job [job...]

atq [-V] [-q queue]

at [-rd] job [job...]

atrm [-V] job [job...]

batch

at -b

Options

at	Executes commands at a specified time.
atq	Lists the user's pending jobs, unless the user is the superuser; in that case, everybody's jobs are listed. The format of the output lines (one for each job) is: Job number, date, hour, queue, and username.
atrm	deletes jobs, identified by their job number.
batch	Executes commands when system load levels permit; in other words, when the load average drops below 1.5, or the value specified in the invocation of atd.

=>at command

shivank@shivank-Vostro-5568:~\$ at now + 2 minute

warning: commands will be executed using /bin/sh

at> echo hello >a.txt

at> <EOT>

job 15 at Sun May 6 12:48:00 2018

shivank@shivank-Vostro-5568:~\$ ls | grep a.txt

shivank@shivank-Vostro-5568:~\$ ls | grep a.txt

a.txt

1:

shivank@shivank-Vostro-5568:~\$ atq

16 Sun May 6 12:52:00 2018 a shivank

shivank@shivank-Vostro-5568:~\$ atrm 16

shivank@shivank-Vostro-5568:~\$ atq

shivank@shivank-Vostro-5568:~\$

Using batch command

```
shivank@shivank-Vostro-5568:~$ batch
warning: commands will be executed using /bin/sh
at> who
at> whoami
at> <EOT>
job 17 at Sun May 6 12:59:00 2018
shivank@shivank-Vostro-5568:~$
```

crontab

Description : Maintain crontab files for individual users (Vixie Cron)

Syntax \$ crontab [-u user] file

crontab [-u user] [-i] { -e | -l | -r }

Options

-e	(edit user's crontab)
-l	(list user's crontab)
-r	(delete user's crontab)
-i	(prompt before deleting user's crontab)

- The asterisk (*) operator specifies all possible values for a field. e.g. every hour or every day.
- The comma (,) operator specifies a list of values, for example: "1,3,4,7,8".
- The dash (-) operator specifies a range of values, for example: "1-6", which is equivalent to "1,2,3,4,5,6".
- The slash (/) operator, can be used to skip a given number of values. For example, "*" / 3 in the hour time field is equivalent to "0,3,6,9,12,15,18,21"; "*" specifies 'every hour' but the "/3" means that only the first, fourth, seventh...and such values given by "*" are used

=>crontab command

```
shivank@shivank-Vostro-5568:~/Documents$ crontab -e
No modification made
shivank@shivank-Vostro-5568:~/Documents$ crontab -e
No modification made
shivank@shivank-Vostro-5568:~/Documents$ crontab -e
crontab: installing new crontab
```

```
shivank@shivank-Vostro-5568:~/Documents$ crontab -l  
*/1 * * * * touch /home/shivank/Documents/a.txt
```


```
shivank@shivank-Vostro-5568:~/Documents$ ls | grep a.txt  
shivank@shivank-Vostro-5568:~/Documents$ ls | grep a.txt  
a.txt  
shivank@shivank-Vostro-5568:~/Documents$ crontab -r  
shivank@shivank-Vostro-5568:~/Documents$ crontab -l  
no crontab for shivank
```

Practical 8

VIM EDITOR

Vim (a contraction of Vi IMproved) is a clone of Bill Joy's vi text editor program for Unix. It was written by Bram Moolenaar based on source for a port of the Stevie editor to the Amiga and first released publicly in 1991. Vim is designed for use both from a command-line interface and as a standalone application in a graphical user interface. Vim is free and open source software and is released under a license that includes some charityware clauses, encouraging users who enjoy the software to consider donating to children in Uganda. The license is compatible with the GNU General Public License through a special clause allowing distribution of modified copies "under the GNU GPL version 2 or any later version".

Although it was originally released for the Amiga, Vim has since been developed to be cross-platform, supporting many other platforms. In 2006, it was voted the most popular editor amongst *Linux Journal* readers



```
VIM - Vi IMproved
          version 7.4.1689
    by Bram Moolenaar et al.
Modified by pkg-vim-maintainers@lists.alioth.debian.org
Vim is open source and freely distributable

Sponsor Vim development!
type  :help sponsor<Enter>    for information

type  :q<Enter>                to exit
type  :help<Enter> or <F1>    for on-line help
type  :help version7<Enter>  for version info
```

Modes in vim editor

Command mode – This mode enables you to perform administrative tasks such as saving the files, executing the commands, moving the cursor, cutting and pasting the lines or words, as well as finding and replacing. In this mode, whatever you type is interpreted as a command.

Command mode has a wide variety of commands and can do things that normal mode can't do as easily. To enter command mode type ':' from normal mode and then type your command which should appear at the bottom of the window.

Insert mode – This mode enables you to insert text into the file. Everything that's typed in this mode is interpreted as input and placed in the file.

Insert commands include:

- **i** for 'insert', this immediately switches vim to insert mode
- **a** for 'append', this moves the cursor after the current character and enters insert mode
- **o** inserts a new line below the current line and enters insert mode on the new line

Moving through the text in command mode

Moving through the text is usually possible with the arrow keys. If not, try:

- **h** to move the cursor to the left
- **l** to move it to the right
- **k** to move up
- **j** to move down

Basic operations

These are some popular **vi** commands:

- **n dd** will delete **n** lines starting from the current cursor position.
- **n dw** will delete **n** words at the right side of the cursor.
- **x** will delete the character on which the cursor is positioned
- **:n** moves to line **n** of the file.
- **:w** will save (write) the file
- **:q** will exit the editor.
- **:q!** forces the exit when you want to quit a file containing unsaved changes.
- **:wq** will save and exit
- **:w newfile** will save the text to **newfile**.
- **:wq!** overrides read-only permission (if you have the permission to override permissions, for instance when you are using the *root* account).

- **/astring** will search the string in the file and position the cursor on the first match below its position.
- **/** will perform the same search again, moving the cursor to the next match.
- **:1, \$s/word/anotherword/g** will replace word with anotherword throughout the file.
- **yy** will copy a block of text.
- **n p** will paste it n times.
- **:recover** will recover a file after an unexpected interruption.

Shell Scripting

A shell script is a computer program designed to be run by the Unix shell, a command-line interpreter. The various dialects of shell scripts are considered to be scripting languages. Typical operations performed by shell scripts include file manipulation, program execution, and printing text.

Logical Operators

Arithmetic Operators

Operator	Description	Example
+ (Addition)	Adds values on either side of the operator	<code>`expr \$a + \$b`</code> will give 30
- (Subtraction)	Subtracts right hand operand from left hand operand	<code>`expr \$a - \$b`</code> will give -10
* (Multiplication)	Multiplies values on either side of the operator	<code>`expr \$a * \$b`</code> will give 200
/ (Division)	Divides left hand operand by right hand operand	<code>`expr \$b / \$a`</code> will give 2
% (Modulus)	Divides left hand operand by right hand operand and returns remainder	<code>`expr \$b % \$a`</code> will give 0
= (Assignment)	Assigns right operand in left operand	<code>a = \$b</code> would assign value of b into a

== (Equality)	Compares two numbers, if both are same then returns true.	[\$a == \$b] would return false.
!= (Not Equality)	Compares two numbers, if both are different then returns true.	[\$a != \$b] would return true.

Relational Operators

Operator	Description	Example
-eq	Checks if the value of two operands are equal or not; if yes, then the condition becomes true.	[\$a -eq \$b] is not true.
-ne	Checks if the value of two operands are equal or not; if values are not equal, then the condition becomes true.	[\$a -ne \$b] is true.
-gt	Checks if the value of left operand is greater than the value of right operand; if yes, then the condition becomes true.	[\$a -gt \$b] is not true.
-lt	Checks if the value of left operand is less than the value of right operand; if yes, then the condition becomes true.	[\$a -lt \$b] is true.
-ge	Checks if the value of left operand is greater than or equal to the value of right operand; if yes, then the condition becomes true.	[\$a -ge \$b] is not true.
-le	Checks if the value of left operand is less than or equal to the value of right operand; if yes, then the condition becomes true.	[\$a -le \$b] is true.

Boolean Operators

Operator	Description	Example
!	This is logical negation. This inverts a true condition into false and vice versa.	[! false] is true.
-o	This is logical OR. If one of the operands is true, then the condition becomes true.	[\$a -lt 20 -o \$b -gt 100] is true.
-a	This is logical AND. If both the operands are true, then the condition becomes true otherwise false.	[\$a -lt 20 -a \$b -gt 100] is false.

Conditional Statements

If statement

```
If condition  
then  
    statements;  
fi
```

If-else statement

```
If condition  
then  
    statements;  
else  
    statements;  
fi
```

If-elif-else statement

```
If condition  
then  
    statements;  
elif condition  
then  
    statements;  
else  
    statements;  
fi
```

While loop

```
while [condition]  
do  
    statements;  
done
```

For loop

```
for { variable name } in { list }  
do  
    statements;  
done
```

Practical 9

Bash Programs for the following Problems

Program to check leap year or not

```
echo "Enter the year :"  
read x  
  
if [ "$(($x % 4))" -eq 0 ] && [ "$(($x % 100))" -ne 0 ]  
then  
    echo "Leap Year"  
else  
    if [ "$(($x % 400))" -eq 0 ]  
    then  
        echo "Leap Year"  
    else  
        echo "Not a Leap Year"  
    fi  
fi
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ sh leapyear.sh  
Enter the year :  
2000  
Leap Year  
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ sh leapyear.sh  
Enter the year :  
1900  
Not a Leap Year
```

Program to check if the number is even or odd

```
echo Enter a number :  
read x  
if [ "$(($x % 2 ))" -ne 0 ]  
then  
    echo "odd"  
else  
    echo "even"  
fi
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ sh evenodd.sh
Enter a number :
15
odd
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ sh evenodd.sh
Enter a number :
20
even
```

Program to find grade on the basis of marks

```
echo enter the marks:
read marks
if [ $marks -gt 90 ]
then
    echo A
elif [ $marks -gt 80 ]
then
    echo B
elif [ $marks -gt 70 ]
then
    echo C
elif [ $marks -gt 60 ]
then
    echo D
elif [ $marks -gt 50 ]
then
    echo E
else
    echo F
fi
```

Output

```
enter the marks:
55
E
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ sh grade.sh
enter the marks:
92
A
```

Program to calculate average marks

```
echo "Enter the number of subjects :"  
read n  
sum=0;  
for sub in $(eval echo {1..$n})  
do  
    echo "Enter the marks in subject $sub :"  
    read marks  
    sum=$((sum + $marks))  
done  
avg=$((sum / $n))  
echo "Average marks : $avg"
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash avgmarks.sh  
Enter the number of subjects :  
3  
Enter the marks in subject 1 :  
85  
Enter the marks in subject 2 :  
90  
Enter the marks in subject 3 :  
95  
Average marks : 90
```

Program to find the circumference of circle

```
echo enter the radius of a circle  
read r  
echo area of circle :  
echo 3.14*$r*$r | bc  
echo perimeter of circle :  
echo 3.14*2*$r | bc
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash circumference.sh  
enter the radius of a circle  
7  
area of circle :  
153.86  
perimeter of circle :  
43.96
```

Practical 10

Bash Programs for the following Problems

Program to find the factorial of a number

```
echo Enter a number :
read x
t=1
y=1
while [ "$y" -le "$x" ]
do
    t=`expr $t \* $y`
    y=`expr $y + 1`
done
echo "the factorial is : $t"
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash factorial.sh
Enter a number :
6
the factorial is : 720
```

Program to print the fibonacci series

```
echo "Enter the desired number of terms required : "
read n
a=1
b=1

if [ $n -ge 1 ]
then
    echo $a
    n=$((n - 1))
fi
if [ $n -ge 1 ]
then
    echo $b
    n=$((n - 1))
fi

for i in $(eval echo {1..$n})
do
    val=$((a + b))
```



```
a=$b
b=$val
echo $val
done
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash fibonacci.sh
```

Enter the desired number of terms required :

```
10
1
1
2
3
5
8
13
21
34
55
```

Program to check if the given number is palindrome or not

```
echo check palindrome
read number
anum=$number
num=0
while [ $number -ne 0 ]
do
num=$((num*10+$number%10))
number=$((number/10))
done
if [ $anum -eq $num ]
then echo it is a palindrome
else
echo not a palindrome
fi
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash pallindrome.sh
```

check palindrome

```
123321
```

it is a palindrome

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash pallindrome.sh
```

check palindrome

12345
not a palindrome

Program to generate prime numbers in the range

```
echo Find Prime number
echo Enter the range
read x1
read x2
echo 'Prime numbers are'
for i in $(seq $x1 1 $x2)
do
j=1
for j in $(seq 2 1 $((i-1)))
do
if [ $((i%j)) -eq 0 ]
then
break
fi
done
if [ $((j+1)) -eq $i ]
then
echo $i
fi
done
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash prime.sh
Find Prime number
Enter the range
5
20
Prime numbers are
5
7
11
13
17
19
```

Program to make calculator using switch case

```

echo calculator
echo enter the 2 numbers
read x1
read x2
echo press 1 for addition
echo press 2 for subtraction
echo press 3 for multiplication
echo press 4 for division
read op
case $op in
1) echo output $((x1+x2));;
2) echo output $((x1-x2));;
3) echo output $((x1*x2));;
4) echo output $((x1/x2));;
*) echo no such operation exist;;
esac

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash calculator.sh
calculator
enter the 2 numbers
15
5
press 1 for addition
press 2 for subtraction
press 3 for multiplication
press 4 for division
4
output 3

```

Program to check if the input is file or directory

```

PASSED=$1
if [ -d "${PASSED}" ]
then echo "${PASSED} is a directory";
elif [ -f "${PASSED}" ]
then echo "${PASSED} is a file";
else echo "${PASSED} is not valid";
    exit 1
fi

```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash fod.sh abcd
abcd is a directory
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash fod.sh fod
fod is not valid
shivank@shivank-Vostro-5568:~/Documents/UE163095/scripts$ bash fod.sh fod.sh
fod.sh is a file
```

Practical 11

Implementation of CPU Scheduling Programs

CPU scheduling is a process which allows one process to use the CPU while the execution of another process is on hold (in waiting state) due to unavailability of any resource like I/O etc, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast and fair.

FCFS

In the "First come first serve" scheduling algorithm, as the name suggests, the process which arrives first, gets executed first, or we can say that the process which requests the CPU first, gets the CPU allocated first.

C++ code

```
#include<iostream>
using namespace std;

struct job{
    int at;
    int bt;
    int pid;
};

bool compare(job a, job b)
{
    return a.at > b.at;
}

void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)
        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}
```

```

void createheap(job *a,int n)
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}

void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}

void input(job *a,int n){
    for(int i=1;i<=n;i++){
        cout<<"Enter arrival time:"<<i<<":";
        cin>>a[i].at;
        cout<<"Enter burst time:"<<i<<":";
        cin>>a[i].bt;
        a[i].pid=i;
    }
}

void output(job *a,int n){
    int time=a[1].at,atat=0,awt=0,tat,wt;
    for(int i=1;i<=n;i++){
        cout<<"Process : "<<a[i].pid;
        cout<<"\tAT:"<<a[i].at;
        if(time<a[i].at)time=a[i].at;
        cout<<"\t ST:"<<time;
        time+=a[i].bt;
        tat=(time-a[i].at);
        cout<<"\tTAT : "<<tat;
        atat+=tat;
        wt=(time-a[i].at-a[i].bt);
        if(wt<0)wt=0;
        cout<<"\tWT : "<<wt;
        awt+=wt;
    }
}

```

```

        cout<<endl;
    }
    cout<<"ATAT:"<<(float)atat/n;
    cout<<"\nAWT:"<<(float)awt/n;
    cout<<"\nThroughput:"<<(float)n/time;
}

void fcfs(job *a,int n){
    heapsort(a,n);
    output(a,n);
}

int main()
{
    int n;
    cout<<"Enter the number of jobs :";
    cin>>n;
    job a[n+1];
    input(a,n);
    fcfs(a,n);
    return 0;
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./fcfs
Enter the number of jobs :5
Enter arrival time:1:0
Enter burst time:1:6
Enter arrival time:2:1
Enter burst time:2:8
Enter arrival time:3:2
Enter burst time:3:0
Enter arrival time:4:3
Enter burst time:4:1
Enter arrival time:5:4
Enter burst time:5:3
Process :1      AT:0      ST:0      TAT :6  WT :0
Process :2      AT:1      ST:6      TAT :13 WT :5
Process :3      AT:2      ST:14     TAT :12 WT :12
Process :4      AT:3      ST:14     TAT :12 WT :11
Process :5      AT:4      ST:15     TAT :14 WT :11
ATAT:11.4
AWT:7.8
Throughput:0.277778shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$

```

SJF:(Shortest Job First)

Shortest Job First, This algorithm associates with each process the length of the process's next CPU burst. When the CPU is available, it is assigned to the process that has the smallest next CPU burst. If the next CPU bursts of two processes are the same, FCFS scheduling is used to break the tie. It is a futuristic process.

SJF (Non-Preemptive)**C++ code**

```
#include<iostream>

using namespace std;

//C++ structure for implementing the jobs for scheduling
struct job{
    int at;          //Arrival time
    int bt;          //Burst time
    int pid; //Process ID
    int done;        //Flag too check if the process is complete or not
};

//Boolean function for comparing two jobs on the basis of arrival time
bool compare(job a, job b)
{
    return a.at > b.at;
}

//Heapify function for heapsort
void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)
        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}

//Create heap for heapsort
void createheap(job *a,int n)
```



```
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}
```

//Function for Heapsort

```
void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}
```

//Function for getting input of all the processes

```
void input(job *a, int n){
    for(int i=1;i<=n;i++)
    {
        cout<<"Enter AT:";
        cin>>a[i].at;
        cout<<"Enter BT:";
        cin>>a[i].bt;
        a[i].pid=i;
        a[i].done=0;
    }
}
```

//Function for displaying the details of the process

```
void output(job a,int wt,int tat){
    cout<<"P-ID : "<<a.pid;
    cout<<"\tWT : "<<wt;
    cout<<"\tTAT : "<<tat<<endl;
}
```

//Function for implementing sjf non pre-emptive

```
void sjf(job *a,int n){
    heapsort(a,n); //Sorting the jobs according to the arrival time
    int time=a[1].at; //Setting up the initial time for managing the processess
    int k=1,wt=0,tat=0,twt=0,ttat=0,min; //Initialising different variables for evaluating AWT
    and ATAT
```

```

a[0].bt=30000; //Infinity value for comparisons
for(int i=1;i<=n;i++) //Loop for selecting the order of jobs
{
    min=0; //Storing of location of job which has arrived and has minimum burst time
    if(i+1>=k && k<=n && time<a[k].at)time=a[k].at;    //if there is some idle time
then to overcome that time
    while(k<=n && a[k].at<=time)k++;    //all the processes with index less than k
has arrived thus the process with minimum burst time is selected in the further for loop
    for(int j=1;j<k;j++){
        if(a[j].bt<a[min].bt && a[j].done==0)
        {
            min=j;
        }
    }
    //All the values are calculated and displayed in the further steps of the function
    a[min].done=1;
    wt=time-a[min].at;
    twt+=wt;
    tat=wt+a[min].bt;
    ttat+=tat;
    output(a[min],wt,tat);
    time+=a[min].bt;
}
cout<<"ATAT: "<<(float)ttat/n<<endl;
cout<<"AWAT: "<<(float)twt/n<<endl;
}

int main()
{
    int n; //The number of processes
    cout<<"Enter the number of the processes :";
    cin>>n;
    job a[n+1];
    input(a,n);    //Getting the details of the jobs
    sjf(a,n);    //scheduling them with sjf
    return 0;
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./sjn_non_preemptive
Enter the number of the processes :5
Enter AT:2
Enter BT:9
Enter AT:8
Enter BT:9
Enter AT:1
Enter BT:2
Enter AT:10
Enter BT:1
Enter AT:56
Enter BT:9
P-ID :3 WT :0   TAT :2
P-ID :1 WT :1   TAT :10
P-ID :4 WT :2   TAT :3
P-ID :2 WT :48  TAT :57
P-ID :5 WT :9   TAT :18
ATAT: 18
AWAT: 12
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ █

```

SJF (Preemptive)**C++ code**

```
#include<iostream>
```

```
using namespace std;
```

```
//C++ structure for implementing the jobs for scheduling
```

```

struct job{
    int at;           //Arrival time
    int bt;           //Burst time
    int tl;           //Time left
    int pid; //Process ID
    int done;         //Flag too check if the process is complete or not
};

```

```
//Boolean function for comparing two jobs on the basis of arrival time
```

```

bool compare(job a, job b)
{
    return a.at > b.at;
}

```

```
//Heapify function for heapsort
```

```

void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)

```

```

        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}

//Create heap for heapsort
void createheap(job *a,int n)
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}

//Funtion for Heapsort
void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}

//Function for getting input of all the processes
void input(job *a, int n){
    for(int i=1;i<=n;i++)
    {
        cout<<"Enter AT:";
        cin>>a[i].at;
        cout<<"Enter BT:";
        cin>>a[i].bt;
        a[i].tl=a[i].bt;
        a[i].pid=i;
        a[i].done=0;
    }
}

```

```

//Function for displaying the details of the process
void output(job a,int tat, int wt,int time){
    cout<<"P-ID : "<<a.pid;
    cout<<"\tWT : "<<wt;
    cout<<"\tTAT : "<<tat;
    cout<<"\tTime : "<<time<<endl;
}

//Function for implementing sjf non pre-emptive
void sjf(job *a,int n){
    heapsort(a,n); //Sorting the jobs according to the arrival time
    int time=a[1].at; //Setting up the initial time for managing the processess
    int k=1,wt=0,tat=0,twt=0,ttat=0,min; //Initialising different variables for evaluating AWT
and ATAT
    a[0].bt=30000; //Infinity value for comparisons
    for(int i=0;i<n;){ //Loop for selecting the order of jobs and count on how many jobs are
completed
        {
            min=0; //Storing of location of job which has arrived and has minimum burst time
            if(i+1>=k && k<=n && time<a[k].at)time=a[k].at; //if there is some idle time
then to overcome that time
            while(k<=n && a[k].at<=time)k++; //all the processes with index less than k
has arrived thus the process with minimum burst time is selected in the further for loop
            for(int j=1;j<k;j++){
                if(a[j].bt<a[min].bt && a[j].done==0)
                {
                    min=j;
                }
            }
        }
        //All the values are calculated and displayed in the further steps of the function

        a[min].tl--; //updating burst time left
        time++; //updating time system time

        if(a[min].tl==0){ //Calculating and displaying the data of completed process
            i++;
            a[min].done=1;
            wt=time-a[min].at-a[min].bt;
            twt+=wt;
            tat=wt+a[min].bt;
            ttat+=tat;
            output(a[min],tat,wt,time);
        }
    }
}

```

```

    }
    cout<<"ATAT: "<<(float)ttat/n<<endl;
    cout<<"AWAT: "<<(float)twat/n<<endl;
}

int main()
{
    int n; //The number of processes
    cout<<"Enter the number of the processes :";
    cin>>n;
    job a[n+1];
    input(a,n); //Getting the details of the jobs
    sjf(a,n); //scheduling them with sjf
    return 0;
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./sjf_preemptive
Enter the number of the processes :5
Enter AT:2
Enter BT:9
Enter AT:8
Enter BT:9
Enter AT:1
Enter BT:2
Enter AT:10
Enter BT:1
Enter AT:56
Enter BT:9
P-ID :3 WT :0   TAT :2   Time :3
P-ID :4 WT :0   TAT :1   Time :11
P-ID :1 WT :2   TAT :11  Time :13
P-ID :2 WT :5   TAT :14  Time :22
P-ID :5 WT :0   TAT :9   Time :65
ATAT: 7.4
AWAT: 1.4
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$

```

Round Robin

Round-robin (RR) is one of the algorithms employed by process and network schedulers in computing. As the term is generally used, time slices (also known as time quanta) are assigned to each process in equal portions and in circular order, handling all processes without priority (also known as cyclic executive). Round-robin scheduling is simple, easy to implement, and starvation-free. Round-robin scheduling can also be applied to other scheduling problems, such as data packet scheduling in computer networks.

C++ Code

```
#include<iostream>
```

```
using namespace std;
```

```
//C++ structure for implementing the jobs for scheduling
```

```
struct job{
    int at;          //Arrival time
    int bt;          //Burst time
    int tl;          //Time left
    int pid; //Process ID
    int done;        //Flag too check if the process is complete or not
};
```

```
//Boolean function for comparing two jobs on the basis of arrival time
```

```
bool compare(job a, job b)
{
    return a.at > b.at;
}
```

```
//Heapify function for heapsort
```

```
void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)
        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}
```

```
//Create heap for heapsort
void createheap(job *a,int n)
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}
```

```
//Funtion for Heapsort
void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}
```

```
//Function for getting input of all the processes
void input(job *a, int n){
    for(int i=1;i<=n;i++)
    {
        cout<<"Enter AT:";
        cin>>a[i].at;
        cout<<"Enter BT:";
        cin>>a[i].bt;
        a[i].tl=a[i].bt;
        a[i].pid=i;
        a[i].done=0;
    }
}
```

```
//Function for displaying the details of the process
void output(job a,int tat, int wt,int time){
    cout<<"P-ID : "<<a.pid;
    cout<<"\tWT : "<<wt;
    cout<<"\tTAT : "<<tat;
    cout<<"\tTime : "<<time<<endl;
}
```

```
//Function for implementing round robin
void RR(job *a,int n){
```



```

    heapsort(a,n); //Sorting the jobs according to the arrival time
    int time=a[1].at; //Setting up the initial time for managing the processes
    int k=1,wt=0,tat=0,twt=0,ttat=0,l=1; //Initialising different variables for evaluating AWT
and ATAT
    for(int i=0;i<n;) //Loop for selecting the order of jobs and count on how many jobs are
completed
    {
        l=l%k;
        if(l==0)l++; //Storing of location of job which has arrived
        if(i+1>=k && k<=n && time<a[k].at)time=a[k].at; //if there is some idle time
then to overcome that time
        while(k<=n && a[k].at<=time)k++; //all the processes with index less than k
has arrived thus the process next that needs to be run is selected in the further for loop
        for(int j=1;j<k;j++,l++){
            l=l%k;
            if(l==0)l++;
            if(a[l].done==0)
            {
                break;
            }
        }
        //All the values are calculated and displayed in the further steps of the function
        a[l].tl--; //updating burst time left
        time++; //updating time
        l++; //shifting to the next job
        if(a[l].tl==0){ //Calculating and displaying the data of completed process
            i++;
            a[l].done=1;
            wt=time-a[l].at-a[l].bt;
            twt+=wt;
            tat=wt+a[l].bt;
            ttat+=tat;
            output(a[l],tat,wt,time);
        }

    }

    cout<<"ATAT: "<<(float)ttat/n<<endl;
    cout<<"AWAT: "<<(float)twt/n<<endl;
}

int main()
{
    int n; //The number of processes
    cout<<"Enter the number of the processes :";

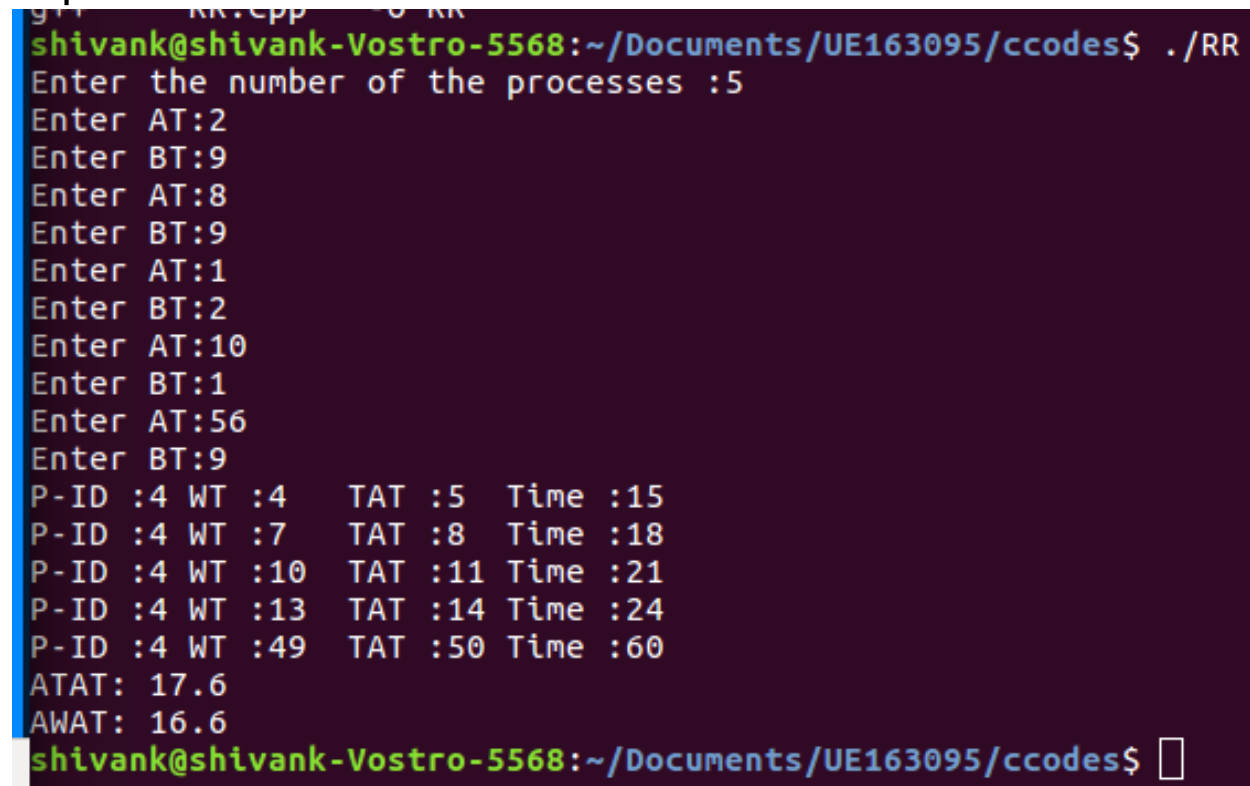
```

```

    cin>>n;
    job a[n+1];
    input(a,n);    //Getting the details of the jobs
    RR(a,n);       //scheduling them with Round Robin
    return 0;
}

```

Output



```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./RR
Enter the number of the processes :5
Enter AT:2
Enter BT:9
Enter AT:8
Enter BT:9
Enter AT:1
Enter BT:2
Enter AT:10
Enter BT:1
Enter AT:56
Enter BT:9
P-ID :4 WT :4   TAT :5   Time :15
P-ID :4 WT :7   TAT :8   Time :18
P-ID :4 WT :10  TAT :11  Time :21
P-ID :4 WT :13  TAT :14  Time :24
P-ID :4 WT :49  TAT :50  Time :60
ATAT: 17.6
AWAT: 16.6
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ 

```

Priority Scheduling

Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems. Each process is assigned a priority. Process with the highest priority is to be executed first and so on.

Processes with the same priority are executed on first come first served basis. Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Priority (Non-preemptive)**C++ Code**

```

#include<iostream>

using namespace std;

//C++ structure for implementing the jobs for scheduling
struct job{
    int at;          //Arrival time
    int bt;          //Burst time
    int pri; //Priority
    int tl;          //Time left
    int pid; //Process ID
    int done;        //Flag too check if the process is complete or not
};

//Boolean function for comparing two jobs on the basis of arrival time
bool compare(job a, job b)
{
    return a.at > b.at;
}

//Heapify function for heapsort
void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)
        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}

//Create heap for heapsort
void createheap(job *a,int n)
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}

```

```

    }
}

//Function for Heapsort
void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}

//Function for getting input of all the processes
void input(job *a, int n){
    for(int i=1;i<=n;i++)
    {
        cout<<"Enter AT:";
        cin>>a[i].at;
        cout<<"Enter BT:";
        cin>>a[i].bt;
        cout<<"Enter Priority :";
        cin>>a[i].pri;
        a[i].tl=a[i].bt;
        a[i].pid=i;
        a[i].done=0;
    }
}

//Function for displaying the details of the process
void output(job a,int tat, int wt,int time){
    cout<<"P-ID : "<<a.pid;
    cout<<"\tWT : "<<wt;
    cout<<"\tTAT : "<<tat;
    cout<<"\tPri : "<<a.pri;
    cout<<"\tTime : "<<time<<endl;
}

//Function for implementing priority pre-emptive
void priority(job *a,int n){
    heapsort(a,n); //Sorting the jobs according to the arrival time
    int time=a[1].at; //Setting up the initial time for managing the processess

```

```

        int k=1,wt=0,tat=0,twt=0,ttat=0,min; //Initialising different variables for evaluating AWT
and ATAT
        a[0].pri=30000; //Infinity value for comparisons
        for(int i=0;i<n;) //Loop for selecting the order of jobs and count on how many jobs are
completed
        {
            min=0; //Storing of location of job which has arrived and has minimum burst time
            if(i+1>=k && k<=n && time<a[k].at)time=a[k].at; //if there is some idle time
then to overcome that time
            while(k<=n && a[k].at<=time)k++; //all the processes with index less than k
has arrived thus the process with maximum priority is selected in the further for loop
            for(int j=1;j<k;j++){
                if(a[j].pri<a[min].pri && a[j].done==0)
                {
                    min=j;
                }
            }
            //All the values are calculated and displayed in the further steps of the function

            a[min].tl--; //updating burst time left
            time++; //updating time system time

            if(a[min].tl==0){ //Calculating and displaying the data of completed process
                i++;
                a[min].done=1;
                wt=time-a[min].at-a[min].bt;
                twt+=wt;
                tat=wt+a[min].bt;
                ttat+=tat;
                output(a[min],tat,wt,time);
            }

        }

        cout<<"ATAT: "<<(float)ttat/n<<endl;
        cout<<"AWAT: "<<(float)twt/n<<endl;
        cout<<"Throughput : "<<(float)n/time<<endl;
    }

int main()
{
    int n; //The number of processes
    cout<<"Enter the number of the processes :";
    cin>>n;
    job a[n+1];

```

```

    input(a,n);    //Getting the details of the jobs
    priority(a,n); //scheduling them with priority
    return 0;
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./Prioritynp
Enter the number of the processes :3
Enter AT:0
Enter BT:10
Enter Priority :0
Enter AT:0
Enter BT:5
Enter Priority :2
Enter AT:0
Enter BT:8
Enter Priority :1
P-ID :1 WT :0   TAT :10 Pri :0
P-ID :3 WT :10  TAT :18 Pri :1
P-ID :2 WT :18  TAT :23 Pri :2
ATAT: 17
AWAT: 9.33333
Throughput :0.130435
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$

```

Priority (Preemptive)

C++ Code

```
#include<iostream>
```

```
using namespace std;
```

```
//C++ structure for implementing the jobs for scheduling
```

```

struct job{
    int at;        //Arrival time
    int bt;        //Burst time
    int pri; //Priority
    int tl;        //Time left
    int pid; //Process ID
    int done;      //Flag too check if the process is complete or not
};

```

```
//Boolean function for comparing two jobs on the basis of arrival time
```

```

bool compare(job a, job b)
{

```

```

        return a.at > b.at;
    }

//Heapify function for heapsort
void heapify(job *a,int &n,int r)
{
    int large=r,left=(2*r),right=(2*r+1);
    if(left<=n)
        if(compare(a[left],a[large]))
            large=left;
    if(right<=n)
        if(compare(a[right],a[large]))
            large=right;
    if(r!=large){
        swap(a[large],a[r]);
        heapify(a,n,large);
    }
}

//Create heap for heapsort
void createheap(job *a,int n)
{
    for(int i=n/2;i>0;i--)
    {
        heapify(a,n,i);
    }
}

//Funtion for Heapsort
void heapsort(job *a,int n)
{
    createheap(a,n);
    while(n>1){
        swap(a[n],a[1]);
        n--;
        heapify(a,n,1);
    }
}

//Function for getting input of all the processes
void input(job *a, int n){
    for(int i=1;i<=n;i++)
    {
        cout<<"Enter AT:";

```

```

        cin>>a[i].at;
        cout<<"Enter BT:";
        cin>>a[i].bt;
        cout<<"Enter Priority :";
        cin>>a[i].pri;
        a[i].tl=a[i].bt;
        a[i].pid=i;
        a[i].done=0;
    }
}

//Function for displaying the details of the process
void output(job a,int tat, int wt,int time){
    cout<<"P-ID : "<<a.pid;
    cout<<"\tWT : "<<wt;
    cout<<"\tTAT : "<<tat;
    cout<<"\tPri : "<<a.pri;
    cout<<"\tTime : "<<time<<endl;
}

//Function for implementing priority pre-emptive
void priority(job *a,int n){
    heapsort(a,n); //Sorting the jobs according to the arrival time
    int time=a[1].at; //Setting up the initial time for managing the processess
    int k=1,wt=0,tat=0,twt=0,ttat=0,min; //Initialising different variables for evaluating AWT
and ATAT
    a[0].pri=30000; //Infinity value for comparisons
    for(int i=0;i<n;){ //Loop for selecting the order of jobs and count on how many jobs are
completed
        {
            min=0; //Storing of location of job which has arrived and has minimum burst time
            if(i+1>=k && k<=n && time<a[k].at)time=a[k].at; //if there is some idle time
then to overcome that time
            while(k<=n && a[k].at<=time)k++; //all the processes with index less than k
has arrived thus the process with maximum priority is selected in the further for loop
            for(int j=1;j<k;j++){
                if(a[j].pri<a[min].pri && a[j].done==0)
                {
                    min=j;
                }
            }
            //All the values are calculated and displayed in the further steps of the function

            a[min].tl--; //updating burst time left

```



```

        time++;           //updating time system time

        if(a[min].tl==0){ //Calculating and displaying the data of completed process
            i++;
            a[min].done=1;
            wt=time-a[min].at-a[min].bt;
            twt+=wt;
            tat=wt+a[min].bt;
            ttat+=tat;
            output(a[min],tat,wt,time);
        }

    }

    cout<<"ATAT: "<<(float)ttat/n<<endl;
    cout<<"AWAT: "<<(float)twt/n<<endl;
    cout<<"Throughput : "<<(float)n/time<<endl;
}

int main()
{
    int n; //The number of processes
    cout<<"Enter the number of the processes :";
    cin>>n;
    job a[n+1];
    input(a,n); //Getting the details of the jobs
    priority(a,n); //scheduling them with priority
    return 0;
}

```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./Priorityp
Enter the number of the processes :3
Enter AT:0
Enter BT:10
Enter Priority :3
Enter AT:1
Enter BT:5
Enter Priority :2
Enter AT:2
Enter BT:2
Enter Priority :1
P-ID :3 WT :0   TAT :2 Pri :1 Time :4
P-ID :2 WT :2   TAT :7 Pri :2 Time :8
P-ID :1 WT :7   TAT :17 Pri :3 Time :17
ATAT: 8.66667
AWAT: 3
Throughput :0.176471
shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$
```

Practical 12

C++ Program to implement Bankers Algorithm

- Safety Algorithm
- Request and Resource Allocation Algorithm

C++ Code

```
#include<iostream>

using namespace std;

struct job{
    int pid;
    int *alo;
    int *max;
    int done;
};

void input(job *a,int n,int m,int *avail){
    for(int i=0;i<n;i++){
        a[i].pid=i;
        a[i].alo=new int[m];
        a[i].max=new int[m];
        a[i].done=0;
        cout<<"Job : "<<i<<endl;
        for(int j=0;j<m;j++){
            cout<<"Allocated : "<<j<<" : ";cin>>a[i].alo[j];
        }
        for(int j=0;j<m;j++){
            cout<<"Max Required : "<<j<<" : ";cin>>a[i].max[j];
        }
    }
    cout<<"Enter details of available ressources : "<<endl;
    for(int j=0;j<m;j++){
        cout<<"Available : "<<j<<" : ";cin>>avail[j];
    }
}

bool compare(job a,int *avail,int m)
{
    for(int i=0;i<m;i++){
        {
            if((a.max[i]-a.alo[i])>avail[i])
                return false;
        }
    }
}
```

```

    }
    return true;
}

void update(job a,int *avail,int m){
    for(int i=0;i<m;i++){
        {
            avail[i]+=a.alo[i];
        }
    }
}

bool Banker(job a[],int avail[],int n, int m){
    int i=0;
    int order[n];
    while(i<n)
    {
        int j=0;
        for(;j<n;j++){
            if(a[j].done==0)
                if(compare(a[j],avail,m)){
                    a[j].done=1;
                    update(a[j],avail,m);
                    order[i]=j;
                    i++;
                    break;
                }
        }
        if(j==n)return false;
    }
    cout<<"Safe state order :"<<endl;
    for(int i=0;i<n;i++)cout<<order[i]<<" ";
    cout<<endl;
    return true;
}

int main()
{
    int n,m;
    cout<<"Enter the number of jobs :";cin>>n;
    cout<<"Enter the number of types of resources :";cin>>m;
    job a[n],b[n];
    int avail[m],ava[m];
    input(a,n,m,avail);
    for(int i=0;i<m;i++){

```

```

        ava[i]=avail[i];
    }
    if(!Banker(a,avail,n,m)){
        cout<<"The system in not in safe state"<<endl;
        return 0;
    }
    int pid,ARR[m];

    for(int i=0;i<m;i++){
        avail[i]=ava[i];
    }

    for(int i=0;i<n;i++){
        a[i].done=0;
    }

    cout<<"Additional Resource Request "<<endl;
    cout<<"Enter process ID:"<<cin>>pid;
    cout<<"Enter the details of required resources :"<<endl;
    for(int j=0;j<m;j++){
        cout<<"REQUIRED : "<<j<<": "<<cin>>ARR[j];
        if(ARR[j]>avail[j]){
            cout<<"Cannot have value greater than available resources\nEnter
Again"<<endl;
            j--;
        }
    }
    for(int i=0;i<m;i++){
        {
            a[pid].alo+=ARR[i];
            avail[i]-=ARR[i];
        }
    }
    if(!Banker(a,avail,n,m)){
        cout<<"The system in not in safe state"<<endl;
        return 0;
    }
    return 0;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./Banker

Enter the number of jobs :5

Enter the number of types of resources :3

Job :0

Allocated :0:0

Allocated :1:1

Allocated :2:0

Max Required :0:7

Max Required :1:5

Max Required :2:3

Job :1

Allocated :0:2

Allocated :1:0

Allocated :2:0

Max Required :0:3

Max Required :1:2

Max Required :2:2

Job :2

Allocated :0:3

Allocated :1:0

Allocated :2:2

Max Required :0:9

Max Required :1:0

Max Required :2:2

Job :3

Allocated :0:2

Allocated :1:1

Allocated :2:1

Max Required :0:2

Max Required :1:2

Max Required :2:2

Job :4

Allocated :0:0

Allocated :1:0

Allocated :2:2

Max Required :0:4

Max Required :1:3

Max Required :2:3

Enter details of available resources :

Available :0:3

Available :1:3

Available :2:2

Safe state order :

1 3 0 2 4

Additional Resource Request

Enter process ID:1

Enter the details of required resources :

REQUIRED :0:1

REQUIRED :1:0

REQUIRED :2:2

The system is not in safe state

Practical 13

Program solve following problems using semaphores

- Producer Consumer Problem
- Reader Writers Problem
- Dining Philosopher Problem

Producer Consumer Problem

C++ Code

```
#include<iostream>
#include<cstdlib>
using namespace std;

class BoundedBuffer
{
    int full, empty, mutex, flagp, flagc; //flag tells if the process is waiting
public:
    BoundedBuffer(int n)
    {
        full = 0;
        empty = n;
        mutex = 1;
        flagp = flagc = 0;
    }
    int wait(int S, int *flag)
    {
        if(S - 1 < 0)
        {
            *flag = 1;
            cout<<"\nWAIT\n";
            return -1;
        }
        else
            return --S;
    }
    int signal(int S)
    {
        return ++S;
    }
    void producer()
    {
```



```

        cout<<"\nItem produced by producer";
        if (wait(empty, &flagp) == -1)
            return;

        empty = wait(empty, &flagp);
        mutex = wait(mutex, &flagp);
        cout<<"\nItem is added to buffer";
        cout<<"\nEmpty buffers: "<<empty;
        mutex = signal(mutex);
        full = signal(full);

        if(flagc == 1)
        {
            cout<<"\n\nConsumer was waiting\n";
            consumer();
            flagc = 0;
        }
    }
}

void consumer()
{
    if (wait(full, &flagc) == -1)
        return;
    full = wait(full, &flagc);
    mutex = wait(mutex, &flagc);
    cout<<"\nItem is removed from buffer";
    cout<<"\nFull buffers: "<<full;
    cout<<"\nItem is consumed by consumer";
    mutex = signal(mutex);
    empty = signal(empty);
    if(flagp == 1)
    {
        cout<<"\n\nProducer was waiting\n";
        producer();
        flagp = 0;
    }
}

};

int main()
{
    int n;
    cout << "Enter the number of buffers ";
    cin >> n;

```

```

    BoundedBuffer b(n);
    char ch;
    int r;
    do
    {
        cout << "\n1. Produce an item\n2.Consume an item\n3.Exit";
        cout << "\nEnter your choice: ";
        cin >> r;
        switch(r)
        {
            case 1:
                b.producer(); break;

            case 2:
                b.consumer(); break;
            case 3:
                exit(0);
        }

        cout << "\nDo you want to continue? ";
        cin >> ch;

    }while(ch == 'y');
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./ProCon
Enter the number of buffers 5

```

```

1. Produce an item
2.Consume an item
3.Exit
Enter your choice: 1

```

```

Item produced by producer
Item is added to buffer
Empty buffers: 4
Do you want to continue? y

```

```

1. Produce an item
2.Consume an item
3.Exit
Enter your choice: 1

```

Item produced by producer
Item is added to buffer
Empty buffers: 3
Do you want to continue? y

1. Produce an item
2. Consume an item
3. Exit
Enter your choice: 2

Item is removed from buffer
Full buffers: 1
Item is consumed by consumer
Do you want to continue? y

1. Produce an item
2. Consume an item
3. Exit
Enter your choice: 2

Item is removed from buffer
Full buffers: 0
Item is consumed by consumer
Do you want to continue? n

Reader Writers Problem

C++ Code

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./readerwriter

1. Begin writing
2. Begin reading
3. End reading
4. Exit
Enter your choice: 1

wrt is 0
Writer is writing now
Do you want to continue? y

1. Begin writing
2. Begin reading
3. End reading
4. Exit

Enter your choice: 2

Reader is reading now

Readers at this moment: 1

Do you want to continue? y

1. Begin writing

2. Begin reading

3. End reading

4. Exit

Enter your choice: 2

Reader is reading now

Readers at this moment: 2

Do you want to continue? y

1. Begin writing

2. Begin reading

3. End reading

4. Exit

Enter your choice: 1

WAIT

Do you want to continue? y

1. Begin writing

2. Begin reading

3. End reading

4. Exit

Enter your choice: 3

Readers at this moment: 1

Do you want to continue? y

1. Begin writing

2. Begin reading

3. End reading

4. Exit

Enter your choice: 4

Dining Philosopher Problem

```

#include<iostream>
#include<cstdlib>
using namespace std;

class DiningPhilosopher
{
    int chopstick[5], flag[5], p[5]; //flag tells if the process is waiting

public:
    DiningPhilosopher()
    {
        for (int i = 0; i < 5; ++i)
        {
            chopstick[i] = 1;
            flag[i] = 0;
            p[i] = 0;
        }
    }

    int wait(int S, int i)
    {
        if(S - 1 < 0)
        {
            flag[i] = 1;
            cout<<"\nWAIT\n";
            return -1;
        }

        else
            return --S;
    }

    int signal(int S)
    {
        return ++S;
    }

    void eat(int i)
    {
        if (wait(chopstick[i], i) == -1)
            return;
    }
}

```

```

        if (wait(chopstick[(i+1)%5], i) == -1)
            return;

    chopstick[i] = wait(chopstick[i], i);
    chopstick[(i+1)%5] = wait(chopstick[(i+1)%5], i);
    cout<<"\n----EAT----";
    p[i] = 1;

    cout << endl;
    for (int j = 0; j < 5; ++j)
        if(p[j] == 1)
            cout << "P" << j+1 <<" ";
    }

void think(int i)
{
    chopstick[i] = signal(chopstick[i]);
    chopstick[ (i+1) % 5 ] = signal(chopstick[(i+1) % 5]);
    p[i] = 0;
    cout<<"\n----THINKING----";

    for (int j = 0; j < 5; ++j)
        if(flag[j] == 1)
        {
            cout<<"\n\nPhilosopher "<< j+1 <<" was waiting\n";
            flag[j] = 0;
            eat(j);
        }
    }

};

int main()
{
    DiningPhilosopher b;

    int r,i;

    do
    {
        cout << "\n1. Eat\n2. Think\n3. Exit";
        cout << "\nEnter your choice: ";
        cin >> r;

```

```

        cout << "\nChoose a philosopher: ";
        cin >> i;
        switch(r)
        {
            case 1:
                b.eat(i-1); break;

            case 2:
                b.think(i-1); break;

            case 3:
                exit(0);
        }
    }while(1);
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./DinPhi

```

1. Eat
2.Think
3.Exit
Enter your choice: 1

```

Choose a philosopher: 1

----EAT----

P1

```

1. Eat
2.Think
3.Exit
Enter your choice: 1

```

Choose a philosopher: 3

----EAT----

P1 P3

```

1. Eat
2.Think
3.Exit
Enter your choice: 1

```

Choose a philosopher: 5

WAIT

- 1. Eat
- 2. Think
- 3. Exit

Enter your choice: 2

Choose a philosopher: 3

----THINKING----

Philosopher 5 was waiting

WAIT

- 1. Eat
- 2. Think
- 3. Exit

Enter your choice: 1

Choose a philosopher: 5

WAIT

- 1. Eat
- 2. Think
- 3. Exit

Enter your choice: 3

Practical 14

Program to implement following Memory Management Algorithms

- MFT
- MVT
 - First Fit
 - Best Fit
 - Worst Fit

MFT

C++ Codes

```
#include<iostream>
#define max 2500
using namespace std;

int main(){
int n=6;
int arr[2][n]={100,150,200,180,350,400},
              { 0, 0, 0, 0, 0, 0}};
int block[100];
int m=0;      int k=0;
for(int i=0;i<n;i++){

    if(m<arr[0][i]){
        m=arr[0][i];
    }
}

int memleft=max;
for (int i=0;i<n;i++)
{

    if(arr[0][i]<=m && arr[1][i]!=1 && memleft>=m){
        memleft-=m;
        arr[1][i]=1;
        block[k]=arr[0][i];k++;
    }
}
for(int i=0;i<k;i++){
    cout<<"Block "<<i<<" Internal Fragmentation : "<<m-block[i]<<endl;
}
if(k<n){
    for(int i=0;i<n;i++)
        if(arr[1][i]!=1)
```

```

        cout<<"PID : "<<i<<" Size : "<<arr[0][i]<<endl;
    }

    cout<<"External fragmentation : "<<memleft;

}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes$ ./mft
Block 0 Internal Fragmentation :300
Block 1 Internal Fragmentation :250
Block 2 Internal Fragmentation :200
Block 3 Internal Fragmentation :220
Block 4 Internal Fragmentation :50
Block 5 Internal Fragmentation :0
External fragmentation :100

```

MVT

C++ Code

```

#include<iostream>
using namespace std;

struct page
{
    int no;
    int size;
};

void FirstFit(int n, page a[], page b[])
{
    int internal;
    cout << "\n-----FIRST FIT-----";
    int i = 0;

    for (int k = 0; k < n; ++k)
    {
        if(i == n)
        {
            cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
            break;
        }
        if((a[i].size - b[k].size) < 0)
        {
            if(i == n-1)
            {
                cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
                break;
            }
        }
    }
}

```

```

        cout<<"\nExternal fragmentation of block"<<b[k].no;
        b[k].size = b[k].size - a[i].size;
        a[i].size = 0;
        i++;
        a[++i].size -= b[k].size;
    }

    else
        a[i].size -= b[k].size;
    i++;
    cout<<"\nBlock "<<b[k].no<<" allocated.";
}

}

//-----BEST FIT-----
void BestFit(int n, page a[], page b[])
{
    int internal = 0;

    cout << "\n-----BEST FIT-----";
    int i = 0;

    //----sorting of b----
    for (i = 0; i < n; ++i)
        for (int j = 0; j < n - i - 1; ++j)
            if(b[j].size > b[j+1].size)
            {
                page t = b[j];
                b[j] = b[j+1];
                b[j+1] = t;
            }

    i = 0;
    for (int k = 0; k < n; ++k)
    {
        if(i == n)
        {
            cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
            break;
        }
        if(a[i].size - b[k].size > 0)
            internal += a[i].size - b[k].size;
        if((a[i].size - b[k].size) < 0)
        {
            if(i == n-1)
            {
                cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
                break;
            }

```

```

    }
    cout<<"\nExternal fragmentation of block"<<b[k].no;
    b[k].size = b[k].size - a[i].size;
    a[i].size = 0;
    i++;
    //a[++i].size -= b[k].size;
}
}
}
//-----WORST FIT-----
void WorstFit(int n, page a[], page b[])
{
    int internal = 0;

    cout << "\n-----WORST FIT-----";
    int i = 0;

    //----sorting of b----
    for (i = 0; i < n; ++i)
        for (int j = 0; j < n - i - 1; ++j)
            if(b[j].size < b[j+1].size)
            {
                page t = b[j];
                b[j] = b[j+1];
                b[j+1] = t;
            }

    i = 0;
    for (int k = 0; k < n; ++k)
    {
        if(i == n)
        {
            cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
            break;
        }

        if(a[i].size - b[k].size > 0)
        {
            internal += a[i].size - b[k].size;
        }
        if((a[i].size - b[k].size) < 0)
        {
            if(i == n-1)
            {
                cout<<"\nBlock "<<b[k].no<<" cannot be allocated. Memory full";
                break;
            }
            cout<<"\nExternal fragmentation of block"<<b[k].no;
            b[k].size = b[k].size - a[i].size;
            a[i].size = 0;

```

```

        i++;
        //a[++i].size -= b[k].size;
    }
    i++;
}
}
//-----MVT-----
void MVT()
{
    int n;
    cout<<"Enter the number of blocks you want to enter: ";
    cin>>n;
    page a[n], c[n], e[n];
    for (int i = 0; i < n; ++i)
    {
        cin >> a[i].size;
        a[i].no = i+1;
    }
    page b[n], d[n], f[n];
    for (int i = 0; i < n; ++i)
    {
        b[i].no = a[i].no;
        c[i].size = a[i].size;
        c[i].no = a[i].no;
        cout<<"\nBlock "<<i+1<<" allocated";
    }
    cout<<"\n\nProgram has been executed. Therefore, all the blocks are empty now. Enter
again\n";
    for (int i = 0; i < n; ++i)
    {
        cin >> b[i].size;
        d[i].size = b[i].size;
        d[i].no = b[i].no;
        f[i].no = b[i].no;
    }
    BestFit(n, a, b);
    cout<<endl;
    FirstFit(n, c, d);
    cout<<endl;
    WorstFit(n, e, f);
}

int main()
{
    MVT();
    cout<<endl;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./mvt

Enter the number of blocks you want to enter: 5

400

200

100

300

500

Block 1 allocated

Block 2 allocated

Block 3 allocated

Block 4 allocated

Block 5 allocated

Program has been executed. Therefore, all the blocks are empty now. Enter again

250

150

200

410

320

-----BEST FIT-----

External fragmentation of block4

-----FIRST FIT-----

Block 1 allocated.

Block 2 allocated.

External fragmentation of block3

Block 3 allocated.

Block 4 cannot be allocated. Memory full

-----WORST FIT-----

External fragmentation of block4

Block 3 cannot be allocated. Memory full

Practical 15

Program to implement the page fault algorithms

- FIFO Page Replacement
- Optimal Page Replacement
- Least Recently Used (LRU)

FIFO Page Replacement

C++ Code

```
#include<iostream>
using namespace std;
#define RANGE 10
int c[3] = {-1,-1,-1};
struct page_array
{
    int page;
    bool valid;
};
void FCFS(page_array *p, int n)
{
    int j, page_faults = 0;
    for (int i = 0; i < n; ++i)
    {
        if(p[i].valid == false)
            cout<<"\nPage is invalid as its valid bit is OFF.";
        else
        {
            for (j = 0; j < 3; ++j)
            {
                if(p[i].page == c[j])
                    break;
            }
            if(j == 3)
            {
                for(int k = 0; k < 2; k++)
                    c[k] = c[k+1];

                c[2] = p[i].page;
                page_faults++;
            }
            cout << endl;
            for(int k = 0; k < 3; k++)
                cout << c[k]<<" ";
        }
    }
    cout << "\n" << page_faults << " page faults occured in FIFO method" <<endl;
}
```

```

int main()
{
    int n;
    cout << "Enter the number of pages you want to load from memory: ";
    cin >> n;
    page_array *p = new page_array [n];
    cout << "\nEnter the page values ";
    for (int i = 0; i < n; ++i)
        cin >> p[i].page;
    for (int i = 0; i < n; ++i)
    {
        if( (p[i].page >= 0) && (p[i].page <= RANGE) )
            p[i].valid = true;
        else
            p[i].valid = false;
    }
    FCFS(p, n);
    return 0;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./FIFOPage

Enter the number of pages you want to load from memory: 20

Enter the page values 1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3

-1 -1 1

-1 1 2

1 2 3

1 2 3

1 2 3

2 3 5

2 3 5

3 5 1

5 1 6

1 6 2

6 2 5

6 2 5

2 5 3

5 3 1

5 3 1

3 1 6

3 1 6

1 6 2

6 2 4

2 4 3

14 page faults occurred in FIFO method

Optimal Page Replacement**C++ Code**

```
#include<stdio.h>
```

```
int main()
{
    int no_of_frames, no_of_pages, frames[10], pages[30], temp[10], flag1, flag2, flag3, i, j, k,
    pos, max, faults = 0;
    printf("Enter number of frames: ");
    scanf("%d", &no_of_frames);
    printf("Enter number of pages: ");
    scanf("%d", &no_of_pages);
    printf("Enter page reference string: ");
    for(i = 0; i < no_of_pages; ++i){
        scanf("%d", &pages[i]);
    }
    for(i = 0; i < no_of_frames; ++i){
        frames[i] = -1;
    }
    for(i = 0; i < no_of_pages; ++i){
        flag1 = flag2 = 0;
        for(j = 0; j < no_of_frames; ++j){
            if(frames[j] == pages[i]){
                flag1 = flag2 = 1;
                break;
            }
        }
        //PAGE FAULT
        if(flag1 == 0){
            for(j = 0; j < no_of_frames; ++j){
                if(frames[j] == -1){
                    faults++;
                    frames[j] = pages[i];
                    flag2 = 1;
                    break;
                }
            }
        }
        if(flag2 == 0){
            flag3 = 0;
            for(j = 0; j < no_of_frames; ++j)
            {
                temp[j] = -1;
                for(k = i + 1; k < no_of_pages; ++k)
                    if(frames[j] == pages[k])
                    {
                        temp[j] = k;
                        break;
                    }
            }
        }
    }
}
```

```

for(j = 0; j < no_of_frames; ++j){
    if(temp[j] == -1){
        pos = j;
        flag3 = 1;
        break;
    }
}
if(flag3 == 0){
    max = temp[0];
    pos = 0;
    for(j = 1; j < no_of_frames; ++j){
        if(temp[j] > max){
            max = temp[j];
            pos = j;
        }
    }
}
frames[pos] = pages[i];
faults++;
}
printf("\n");
for(j = 0; j < no_of_frames; ++j){
    printf("%d\t", frames[j]);
}
}
printf("\n\nTotal Page Faults = %d", faults);

return 0;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./OPR

Enter number of frames: 3

Enter number of pages: 20

Enter page reference string: 1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3

1	-1	-1
1	2	-1
1	2	3
1	2	3
1	2	3
1	2	5
1	2	5
1	2	5
6	2	5
6	2	5
6	2	5
6	2	5

6	2	3
6	1	3
6	1	3
6	1	3
6	1	3
2	1	3
4	1	3
4	1	3

Total Page Faults = 9

Least Recently Used

C++ Code

```
#include<stdio.h>
```

```
int findLRU(int time[], int n){
    int i, minimum = time[0], pos = 0;
```

```
    for(i = 1; i < n; ++i){
        if(time[i] < minimum){
            minimum = time[i];
            pos = i;
        }
    }
}
```

```
    return pos;
}
```

```
int main()
{
    int no_of_frames, no_of_pages, frames[10], pages[30], counter = 0, time[10], flag1, flag2, i, j,
    pos, faults = 0;
    printf("Enter number of frames: ");
    scanf("%d", &no_of_frames);

    printf("Enter number of pages: ");
    scanf("%d", &no_of_pages);

    printf("Enter reference string: ");

    for(i = 0; i < no_of_pages; ++i){
        scanf("%d", &pages[i]);
```

```
}

for(i = 0; i < no_of_frames; ++i){
    frames[i] = -1;
}
for(i = 0; i < no_of_pages; ++i){
    flag1 = flag2 = 0;

    for(j = 0; j < no_of_frames; ++j){
        if(frames[j] == pages[i]){
            counter++;
            time[j] = counter;
            flag1 = flag2 = 1;
            break;
        }
    }

    if(flag1 == 0){
        for(j = 0; j < no_of_frames; ++j){
            if(frames[j] == -1){
                counter++;
                faults++;
                frames[j] = pages[i];
                time[j] = counter;
                flag2 = 1;
                break;
            }
        }
    }

    if(flag2 == 0){
        pos = findLRU(time, no_of_frames);
        counter++;
        faults++;
        frames[pos] = pages[i];
        time[pos] = counter;
    }

    printf("\n");
    for(j = 0; j < no_of_frames; ++j){
        printf("%d\t", frames[j]);
    }
}

printf("\n\nTotal Page Faults = %d", faults);
```

```
    return 0;  
}
```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/ccodes\$./LRU

Enter number of frames: 3

Enter number of pages: 20

Enter reference string: 1 2 3 2 1 5 2 1 6 2 5 6 3 1 3 6 1 2 4 3

1	-1	-1
1	2	-1
1	2	3
1	2	3
1	2	3
1	2	5
1	2	5
1	2	5
1	2	6
1	2	6
5	2	6
5	2	6
5	3	6
1	3	6
1	3	6
1	3	6
1	3	6
1	2	6
1	2	4
3	2	4

Total Page Faults = 11

Practical 16

Program to implement the following Disk Scheduling Algorithms

- FCFS (First Come First Serve)
- SSFT (Shortest Seek Time First)
- Scan
- C-Scan
- Look
- C-Look

FCFS

C++ Code

```
#include<iostream>
#include<cmath>
using namespace std;

int FCFS(int a[],int n, int ptr){
    int val=0;
    for(int i=0;i<n;i++)
    {
        val+=abs(ptr-a[i]);
        ptr=a[i];
    }
    return val;
}

int main()
{
    int n;
    cout<<"Enter n :";cin>>n;
    int a[n],ptr;
    cout<<"Enter the memory locations :\n";
    for(int i=0;i<n;i++)
        cin>>a[i];
    cout<<"Enter initial ptr location :";cin>>ptr;
    cout<<"Total Head Movement :"<<FCFS(a,n,ptr);
    return 0;
}
```

Output

```
shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling$ ./FCFS
Enter n :5
Enter the memory locations :
```

23
 89
 132
 42
 187
 Enter initial ptr location :100
 Total Head Movement :421

SSFT (Shortest Seek Time First)

```
#include<iostream>
#include<algorithm>
#include<cmath>
using namespace std;

int a[2][100];

int prev(int i){
    while(a[1][i]==1&& i>-1) i--;
    return i;
}

int next(int i,int n){
    while(a[1][i]==1&& i<n) i++;
    return i;
}

int SSTF(int n, int ptr){
    int val=0;
    sort(a[0],a[0]+n);
    int i=0;
    while(a[0][i]<ptr) i++;
    if(i==0||abs(a[0][i]-ptr)<abs(a[0][i-1]-ptr))
    {
        val=abs(a[0][i]-ptr);
        a[1][i]=1;
        ptr=a[0][i];
    }
    else{
        val=abs(a[0][i-1]-ptr);
        a[1][i-1]=1;
        ptr=a[0][i-1];
        i--;
    }
}
```

```

    int k,l;
    for(int j=1;j<n;j++)
    {
        k=prev(i);
        l=next(i,n);
        if(k== -1)i=l;
        else if(l==n)i=k;
        else if(abs(a[0][k]-ptr)>abs(a[0][l]-ptr))i=l;
        else i=k;
        a[1][i]=1;
        val+=abs(ptr-a[0][i]);
        ptr=a[0][i];
    }
    return val;
}

int main()
{
    int n,ptr;
    cout<<"Enter n :";cin>>n;
    cout<<"Enter the memory locations :\n";
    for(int i=0;i<n;i++)
    {
        cin>>a[0][i];
        a[1][i]=0;
    }
    cout<<"Enter initial ptr location :";cin>>ptr;
    cout<<"Total Head Movement :"<<SSTF(n,ptr);
    return 0;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling\$./SSTF

Enter n :5

Enter the memory locations :

23

89

132

42

187

Enter initial ptr location :100

Total Head Movement :273

SCAN**C++ Code**

```
#include <iostream>
#include<cmath>
using namespace std;

int main()
{
    int n,d[8],a,b,c=0,j,i=0;
    char ch='y';
    cout<<"Enter no. of elements in queue : ";
    cin>>n;
    cout<<"enter value of initial head position : ";
    cin>>a;

    for(i=0;i<n;i++)
    {
        cout<<"Enter the memory locations "<<i+1<<" : ";
        cin>>d[i];
    }

    for(i=0;i<n;i++)
    for(j=0;j<n;j++)
    {
        if(d[i]<d[j])
        {
            b=d[i];
            d[i]=d[j];
            d[j]=b;
        }
    }

    for(i=0;i<n;i++)
    {
        if(d[i]>a)
        {
            j=i;
            break;
        }
    }

    c=0;
    b=0;
```

```

do
{
    c+=abs(b-d[j]);
    b=d[j];
    j++;
}while(j<n);

c=c+a;
cout<<" \nTotal head movement = "<<c<<" cylinders";
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling\$./scan

Enter no. of elements in queue : 5

enter value of initial head position : 100

Enter the memory locations 1 : 23

Enter the memory locations 2 : 89

Enter the memory locations 3 : 132

Enter the memory locations 4 : 42

Enter the memory locations 5 : 187

Total head movement = 287 cylinders

C-Scan

C++ Code

```
#include<bits/stdc++.h>
```

```
using namespace std;
```

```
int main()
```

```
{
```

```
    int num;
```

```
    cout<<"Number of memory locations :";
```

```
    cin>>num;
```

```
    int arr[num];
```

```
    cout<<"Enter Locations :";
```

```
    for(int i = 0; i < num; i++)
```

```
        cin>>arr[i];
```

```
    int head;
```

```
    cout<<"Enter starting position of head - ";
```

```
    cin>>head;
```

```

int max;
cout<<"Enter maximum number of cylinnders : ";
cin>>max;

int cur = 0;
for(int i = 0; i < num; i++)
{
    if(arr[i] > cur && arr[i] <= head)
        cur = arr[i];
}
int movement = 2 * max - head + cur;
cout<<"Total movement - "<<movement;
}

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling$ ./cscan
Number of memory locations :8
Enter Locations :100 175 51 133 8 140 73 77
Enter starting position of head - 63
Enter maximum number of cylinders : 199
Total movement - 386

```

Look

C++ Code

```

#include<iostream>
#include<algorithm>
using namespace std;

int main()
{
    int max, arr[20], cur, m, sum=0,prev, index;
    cout<<"Enter total number of cyllinders:"<<endl;
    cin>>max;
    cout<<"Enter the number of locations to be searched"<<endl;
    cin>>m;
    cout<<"Enter the current position: "<<endl;
    cin>>cur;
    cout<<"Enter the previous position: "<<endl;
    cin>>prev;
    cout<<"Enter the values"<<endl;
    for(int i=0; i<m; i++)
        cin>>arr[i];
}

```

```

sort(arr , arr+m);
for(int i=0 ; i<m; i++)
{
    if (cur < arr[i] && cur > prev)
    {
        index = i;
        break;
    }
    if( cur < arr[i] && cur < prev)
    {
        index = i-1;
        break;
    }
}
if(cur > prev){
    for(int i=index; i<m; i++)
    {
        sum += arr[i] - cur;
        cur = arr[i];
    }
    sum += (arr[m-1] - arr[index-1]);
    cur = arr[index-1];
    for(int i=index-1; i>=0 ; i--)
    {
        sum += cur - arr[i];
        cur = arr[i];
    }
}
if(cur < prev)
{
    for(int i=index; i>=0; i--)
    {
        sum += cur - arr[i];
        cur = arr[i];
    }
    sum += (arr[index+1] - arr[0]);
    cur = arr[index+1];
    for(int i=index+1; i<m ; i++)
    {
        sum += arr[i] - cur;
        cur = arr[i];
    }
}
cout<<"Total number of head moments: "<<sum;

```

```

        return 0;
    }

```

Output

```

shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling$ ./look
Enter total number of cylinders:
200
Enter the number of locations to be searched
5
Enter the current position:
100
Enter the previous position:
110
Enter the values
23
89
132
42
187
Total number of head moments: 241

```

C-Look

C++ Code

```

#include<iostream>
#include<algorithm>
using namespace std;

int main()
{
    int max, arr[20], cur, m, sum=0,prev, index;
    cout<<"Enter total number of cylinders:"<<endl;
    cin>>max;
    cout<<"Enter the number of locations to be searched"<<endl;
    cin>>m;
    cout<<"Enter the current position: "<<endl;
    cin>>cur;
    cout<<"Enter the previous position: "<<endl;
    cin>>prev;
    cout<<"Enter the values"<<endl;
    for(int i=0; i<m; i++)
        cin>>arr[i];
    sort(arr , arr+m);
    for(int i=0 ; i<m; i++)

```

```

{
    if (cur < arr[i] && cur > prev)
    {
        index = i;
        break;
    }
    if( cur < arr[i] && cur < prev)
    {
        index = i-1;
        break;
    }
}
if(cur > prev){
    for(int i=index; i<m; i++)
    {
        sum += arr[i] - cur;
        cur = arr[i];
    }
    sum += (arr[m-1] - arr[0]);
    cur = arr[0];
    for(int i=0; i < index ; i++)
    {
        sum += arr[i] - cur;
        cur = arr[i];
    }
}
if(cur < prev)
{
    for(int i=index; i>=0; i--)
    {
        sum += cur - arr[i];
        cur = arr[i];
    }
    sum += (arr[m-1] - arr[0]);
    cur = arr[m-1];
    for(int i=m-1; i>=index+1 ; i--)
    {
        sum += cur - arr[i];
        cur = arr[i];
    }
}
cout<<"Total number of head moments: "<<sum;
return 0;
}

```

Output

shivank@shivank-Vostro-5568:~/Documents/UE163095/Disk Sheduling\$./clook

Enter total number of cylinders:

200

Enter the number of locations to be searched

5

Enter the current position:

100

Enter the previous position:

110

Enter the values

23

89

132

42

187

Total number of head moments: 296s